

ABSTRACT

Title of Dissertation: EVALUATING MACHINE INTELLIGENCE
WITH QUESTION ANSWERING

Pedro Rodriguez, 2021
Doctor of Philosophy, 2021

Dissertation directed by: Professor Jordan Boyd-Graber
Department of Computer Science
College of Information Studies
Language Science Center
Institute for Advanced Computer Studies

Humans ask questions to learn about the world and to test knowledge understanding. The ability to ask questions combines aspects of intelligence unique to humans: language understanding, knowledge representation, and reasoning. Thus, building systems capable of intelligent question answering (QA) is a grand goal of natural language processing (NLP). To measure progress in NLP, we create “exams” for computer systems and compare their effectiveness against a reference point—often based on humans. How precisely we measure progress depends on whether we are building computer systems that optimize human satisfaction in information-seeking tasks or that measure progress towards intelligent QA. In the first part of this dissertation, we explore each goal in turn, how they differ, and describe their relationship to QA formats. As an example of an information-seeking evaluation, we introduce a new dialog QA task paired with a new evaluation method. Afterward, we turn our attention to using QA to evaluate machine intelligence.

A good evaluation should be able to discriminate between lesser and more capable QA models. This dissertation explores three ways to improve the discriminative power of QA evaluations: (1) dynamic weighting of test questions, (2) a format that by construction tests multiple levels of knowledge, and (3) evaluation data that is created through human-computer collaboration.

By dynamically weighting test questions, we challenge a foundational assumption of the de facto standard in QA evaluation—the leaderboard. Namely, we contend that contrary to nearly all QA and NLP evaluations which implicitly assign equal weights to examples by averaging scores, that examples are *not* equally useful for estimating machine (or human) QA ability. As any student may tell you, not all questions on an exam are equally difficult and in the worst-case questions are unsolvable. Drawing on decades of research in educational testing, we propose adopting an alternative evaluation methodology—Item Response Theory—that is widely used to score human exams (e.g., the SAT). By dynamically weighting questions, we show that this improves the reliability of leaderboards in discriminating between models of differing QA ability while also being helpful in the construction of new evaluation datasets.

Having improved the scoring of models, we next turn to improving the format and data in QA evaluations. Our idea is simple. In most QA tasks (e.g., Jeopardy!), each question tests a single level of knowledge; in our task (the trivia game Quizbowl), we test multiple levels of knowledge with each question. Since each question tests multiple levels of knowledge, this decreases the likelihood that we learn nothing about the difference between two models (i.e., they are both correct or both

wrong), which substantially increases discriminative power.

Despite the improved format, we next show that while our QA models defeat accomplished trivia players, that they are overly reliant on brittle pattern matching, which indicates a failure to *intelligently* answer questions. To mitigate this problem, we introduce a new framework for building evaluation data where humans and machines cooperatively craft trivia questions that are difficult to answer through clever pattern matching tricks alone—while being no harder for humans.

We conclude by sketching a broader vision for QA evaluation that combines the three components of evaluation we improve—scoring, format, and data—to create living evaluations and re-imagine the role of leaderboards.

EVALUATING MACHINE INTELLIGENCE
WITH QUESTION ANSWERING

by

Pedro Rodriguez

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2021

Advisory Committee:

Professor Jordan Boyd-Graber, Chair

Professor Douglas W. Oard (Dean's Representative)

Professor Leilani Battle

Professor Leo Zhicheng Liu

Professor John P. Lalor

© Copyright by
Pedro Rodriguez
2021

Dedication

In memory of the Apollo 1, Challenger, and Columbia crews. Their bravery and dedication to exploration, discovery, and science has—and always will—inspire me.

Virgil I. Grissom	Francis R. Scobee	Rick Husband
Edward H. White II	Michael J. Smith	William C. McCool
Roger B. Chaffee	Ronald McNair	Michael P. Anderson
	Ellison Onizuka	Kalpana Chawla
	Judith Resnik	David M. Brown
	Gregory Jarvis	Laurel Clark
	Christa McAuliffe	Ilan Ramon

“From our orbital vantage point, we observe an Earth without borders, full of peace, beauty and magnificence, and we pray that humanity as a whole can imagine a borderless world as we see it, and strive to live as one in peace.”

— William C. McCool, January 29, 2003

Acknowledgments

I am deeply grateful to all my mentors, colleagues, friends, and family for their nurturing, love, and encouragement over the years. Their support helped me grow as a researcher, craft this dissertation, and achieve my dream.

First, I would like to thank my advisor, Jordan Boyd-Graber, for his steady guidance through graduate school. Although at first things like abiding by a book-length style guide seemed arcane, through time I came to appreciate and (attempt to) adopt the same attention to detail in all my work. Among the many things Jordan taught me, I am especially grateful for his guidance in improving as a writer, by extension improving as a critical thinker, and emphasis on digging deep into problems and literature. I am also grateful for the freedom Jordan gave me to try research ideas, the patience to let me learn from my mistakes and failed projects, and the confidence in me when I tried a new idea. This helped me discover and explore my interests while building an understanding of how to conduct research. For this and the countless other things I learned via osmosis, thank you.

Thank you as well to the members of my dissertation committee. Doug's uncanny ability to ask simple, yet deep-cutting fundamental questions was crucial in helping me develop and improve several ideas. John's expertise and guidance in item response theory helped me apply it to question answering evaluation, and I'm proud of the resulting paper. Both Leilani's proposal feedback and Leo's visualization class helped inspire the user-centered perspective I take in laying out future work.

Being part of the CLIP community is among the best parts of my PhD experience.

rience. Through CLIP, I met many collaborators, friends, and colleagues. Among the wonderful individuals I'd like to thank are: Yogarshi Vyas, Joe Barrow and Han-Chin Shing for all the board game nights, bottomless boba tea, and reflections on being a PhD student; Joe Barrow for introducing me to the wonderful biking around College Park—even that one time we accidentally went mountain biking on road bikes; Shi Feng for being a friend and collaborator that always seems to ask the right questions; and many former and current CLIP members including Mohit Iyyer, Alvin Grissom II, Eric Wallace, Kianté Brantley, Pranav Goel, Peter Rankel, Alexander Hoyle, Matthew Shu, and Fenfei Guo. Finally, thank you to the CLIP faculty for creating and nurturing such an amazing community that I am proud to be a part of.

Although I completed my PhD at UMD, my adventure began at CU in beautiful Boulder, Colorado. In those early years, I fell in love with the boundless outdoors and am grateful for the time I had to explore Colorado's skiing, rock climbing, and hiking. In my time there, I met several friends and mentors who I'd like to thank: Dirk Grunwald for long discussions on anything from hardware to research to backcountry skiing; Rafael Frongillo for supporting the Data Science Team which, in retrospect, likely influenced my later interest in leaderboards; Davis Yoshida for intense and insightful research discussions; Alvin Grissom II for further opening my eyes to racial injustices and giving me the opportunity to become involved in service to the ACL community. At both CU Boulder and UMD, I had amazing graduate program advisors in Rajshree Shrestha and Tom Hurst.

Through internships, I was fortunate to work with fantastic industry mentors

whose diverse perspectives taught me different ways to frame, approach, think about, and solve research problems. At Riot Games, I am grateful to Xiaoyang Yang, Wes Kerr, Ben Kasper, and Kimberly Voll for pushing me to think hard about the expected impacts of a potential project *before* diving head-first into it. At Google, I had many insightful discussions with Jannis Bulian and Benjamin Börschinger.

When working in the engineering trenches of a research project, a mistake I often struggled with—especially early on—was losing sight of the science. Although I’m sure he wasn’t the first or only person to point this out to me, Paul Crook (Facebook Research) gave me a firm, but gentle reminder, at the right moment, of something I once knew: that a critical part of science lies in making worthwhile, testable hypotheses and then devising ways to test them. This small nudge eventually blossomed into the best parts of my EMNLP internship paper. In addition to Paul, thank you to Shane Moon, Stephen Wang, and Becca Silvert.

Throughout this journey, close friends and family have been an ever-present source of loving support through the best and worst of times. To Stan and Kasandra Bessey, thank you for all the ways, small and large, you’ve unconditionally supported me. To my god-mother Pearline, thank you for your unconditional love and sharing your joy for life with not just me, but generations of my family. To my dad Chago, thank you for nurturing my curiosity and love for science while kindling my desire for exploration through the outdoors. To my mom Lourdes, thank you for your boundless and unconditional love, always believing in me, and being there when I needed you most. To my brother Fritz, you’ve always (lovingly) been my most honest critic; more importantly though, you’ve always been there for me.

Table of Contents

Dedication	ii
Acknowledgements	iii
List of Tables	xi
List of Figures	xv
1 Introduction	1
1.1 Overview	1
1.2 Conversational Information-Seeking	2
1.3 Questions are Not Equally Informative of Ability	3
1.4 Incremental QA for Polytomous Evaluation of Knowledge	3
1.5 Crafting Robust Questions Through Cooperative Machine-Human Authoring	5
1.6 Roadmap	6
2 Background	7
2.1 The Epistemological Heritage of Question Answering	7
2.1.1 The Cranfield Paradigm	8
2.1.2 The Manchester Paradigm	9
2.1.3 Common Ground	12
2.2 Question Answering Dataset Characteristics	13
2.2.1 Answer Formats	14
2.2.2 Information Context	16
2.2.3 Generative Story	17
2.3 Question Answering Datasets	21
2.4 Methods for Question Answering	23
2.4.1 A Brief History of Symbolic Question Answering Models	23
2.4.2 Modern Methods for Question Answering	24
2.4.3 Neural Text Encoders	26
2.4.4 Document Retrieval	28
2.4.5 Neural Document Retrieval	29

2.4.6	Neural Answering Module	30
2.5	Prior and Related Work in Quizbowl	31
2.6	The Harrowing Past of Measuring Human Intelligence	33
3	Information Seeking in the Spirit of Learning:	
	A Dataset for Conversational Curiosity	35
3.1	Motivation for Conversational Information-Seeking	36
3.2	Building the Curiosity Dataset	38
3.2.1	Choosing the Geographic Topics, Aspects, and Facts for the Dataset	38
3.2.2	User and Assistant Dialog Interfaces	38
3.2.3	Dialog Act Annotation	40
3.2.4	Validating Data Quality	41
3.3	Dataset Analysis	42
3.3.1	What Facts do User Prefer?	43
3.3.1.1	Likes for Explicit Preference Elicitation	44
3.3.1.2	Mining Acts for Implicit Preferences	44
3.3.1.3	Paraphrase Analysis	44
3.3.2	Dataset Statistics	44
3.4	Models	44
3.4.1	Text Representation	45
3.4.2	Dialog Representation	45
3.4.3	Tasks and Loss Functions	47
3.4.4	Model Implementation and Training Details	48
3.5	Modeling Experiments	49
3.5.1	Baselines	49
3.5.2	Discussion	49
3.6	Related Work	50
3.7	Future Work and Conclusion	52
4	Evaluation Examples are not Equally Informative: How should that change NLP Leaderboards?	54
4.1	Leaderboards are Shiny	54
4.1.1	How to Direct Leaderboards' Light	56
4.2	A Generative Story for Leaderboards	56
4.2.1	Examples are Not Equally Useful	58
4.2.2	Inference	59
4.3	Ranking and Comparing Subjects	59
4.4	IRT for Leaderboards	60
4.4.1	Why a Linear Model Baseline	60
4.4.2	Response Prediction is Accurate	60
4.4.2.1	SQuAD Leaderboard Data	61
4.4.2.2	Evaluation Scheme	61
4.4.2.3	IRT Response Prediction is Accurate	61
4.4.2.4	What Model Features are Predictive?	62

4.4.3	Ranking with IRT	62
4.4.3.1	IRT Rankings Have Better Reliability	62
4.4.4	Statistical Significance of Difference in Kendall Tau Coefficients	63
4.4.5	IRT Improves Cold Start Reliability	64
4.5	Qualitative Insights on Leaderboards	66
4.5.1	Guiding Analysis with IRT	66
4.5.2	Identifying Annotation Error	67
4.6	Related Work	67
4.7	Conclusion	70
4.8	Limitations	70
4.9	Future Work	70
5	The Case for Incremental Question Answering Evaluation	71
5.1	An Introduction to Quizbowl	72
5.2	Why Quizbowl?	73
5.2.1	What is a Buzzer Race?	74
5.2.2	Pyramidality and Buzzers	75
5.2.3	The Craft of Question Writing	76
5.2.4	Quizbowl for Natural Language Processing Research	78
5.2.5	Quizbowl for Machine Learning Research	79
5.3	The QANTA Dataset	81
5.3.1	Sources of Quizbowl Questions	81
5.3.2	QANTA Questions	81
5.3.3	Gameplay Records: Recording Humans Playing Quizbowl Online	83
5.3.4	Preprocessing	85
5.4	Deciding When and What to Answer	88
5.5	Guessing QB Answers	90
5.5.1	Explicit Pattern Matching with Information Retrieval	90
5.5.2	Trainable Pattern Matching with Linear Models	91
5.5.3	Neural Network Models	91
5.6	Buzzing	94
5.6.1	A Classification Approach to Buzzing	95
5.7	Offline Evaluation	96
5.7.1	Evaluating the Guesser	97
5.7.2	Identifying Sources of Error	99
5.7.3	Evaluating the Buzzer	103
5.8	Live Exhibition Events	105
5.9	Related Work	107
5.9.1	Question Answering Datasets	108
5.9.2	Answer Triggering and Model Calibration	110
5.9.3	Opponent Modeling	110
5.10	Future Work	110
5.10.1	Generalization in Factoid Question Answering	111
5.10.2	Robust, Trustable, and Explainable Machine Learning	112

5.11	Conclusion	112
6	Centaur Authoring of Adversarial Questions	114
6.1	Introduction	114
6.2	Adversarial Evaluation for NLP	115
6.2.1	Putting a Human in the Loop	116
6.3	Our QA Testbed: Quizbowl	116
6.3.1	Known Exploits of Quizbowl Questions	116
6.3.2	Models and Datasets	117
6.3.3	Interpreting Quizbowl Models	117
6.3.4	Adversarial Writing Interface	118
6.3.5	Question Authors	119
6.3.6	How an Author Writes a Question	119
6.4	A New Adversarially-Authored Dataset	120
6.4.1	Validating Questions with Quizbowlers	120
6.5	Computer Experiments	121
6.5.1	First Round Attacks: IR Adversarial Questions Transfer To All Models	121
6.5.2	Second Round Attacks: RNN Adversarial Questions are Brittle	121
6.5.3	Humans vs. Computer, Live (again)!	122
6.6	What Makes Adversarially-authored Questions Hard?	124
6.6.1	Quantitative Differences in Questions	124
6.6.2	Categorizing Adversarial Phenomena	124
6.6.3	Adversarial Category 1: Reasoning	125
6.6.4	Adversarial Category 2: Distracting Clues	125
6.7	How Do Interpretations Help?	127
6.7.1	Interviews With Adversarial Authors	129
6.8	Related Work	129
6.9	Conclusion	130
7	Conclusion	131
7.1	Future Work: Living Evaluations	133
7.1.1	Incorporating Content Models into IRT Models	133
7.1.2	Guiding Example Creation	134
7.1.3	Tender Loving Care for Underperforming Examples	135
7.1.4	Multi-Examples	135
7.1.5	Effects of Continually Updating Evaluations	135
7.1.6	Model Cards	136
7.2	What More Should Leaderboards Do?	136
7.2.1	Stakeholders	137
7.2.2	Ranking View	139
7.2.3	Model View	141
7.2.4	Example View	142
7.3	Future Directions in Item Response Theory for NLP	142
7.3.1	Multidimensional Clustering	142

7.3.2	Statistical Testing	143
7.4	Reflections and Synthesis	144
A	Curiosity	146
A.1	Components of Dialog Interfaces	146
A.1.1	User’s Interface	146
A.1.2	Assistant Interface	147
A.2	Dialog Act Annotation	148
A.3	Sample Dialogs	148
A.4	Paraphrase Analysis and Samples	149
A.5	Like Prediction Comparison	149
A.6	Model Training, Implementation, and Computation	149
A.7	MS Marco Conversational Sample Queries	149
B	Leaderboard	155
B.1	SQuAD Item Examples	155
B.2	Logistic Regression Features	155
B.3	IRT Model Type Correlation	155
B.4	Ranking Stability Experiments	155
B.4.1	Development and Test Set Correlations	156
B.5	The IRT Statistical Test	156
B.6	Multidimensional IRT Clustering	157
B.7	Reproducibility Checklist	158
B.7.1	Software and Parameters	158
B.7.2	Hyperparameters	159
B.7.3	Computational Resources	160
C	Quizbowl	162
C.1	Preprocessing	162
C.1.1	Aligning and De-duplicating Questions	162
C.1.2	Textual Preprocessing	162
C.1.3	Fold Assignment	162
C.1.4	Matching QB Answers to Wikipedia Pages	163
C.1.5	Buzzer features	165
C.2	Natural Questions Categories	166
D	Centaur Authorship of Quizbowl Questions	167
D.1	Failure of Syntactically Controlled Paraphrase Networks	167
D.2	Studio Ousia QB Model	168

List of Tables

2.1	The table categorizes datasets by the task’s paradigm, the Author (human 🧑, machine 🤖, or a mix), and topic domain. To be categorized as human-generated, it is not enough for the dataset to consist of naturally occurring text; it must have questions that are authored by humans instead of being automatically generated. 🧑→🤖 and 🧑→🤖 indicate human-generated questions that are non-trivially checked or modified by a machine system. 🤖→🧑 and 🤖→🧑 indicate machine-generated questions that are paraphrased by humans. 🧑↔️🤖 and 🧑↔️🤖 indicate question creation incorporates interactive human-machine cooperation.	23
3.1	Counts, descriptions and examples of the dataset’s dialog acts. . . .	42
3.2	Consider a task where each utterance has labels A and B. In the single-label version, each utterance is labeled as either A or B. The table shows the outcome of converting the multi-label version to single-label by creating a row for each example-label combination. Cell values are binary indicators.	43
3.3	We analyze the paraphrases annotators use through manual categorization. The “Copy” category includes cherry-picked verbatim phrases, verbatim copies, and contextualized copies (e.g., changing a named entity to “it”). The majority of paraphrases are correct and only incorporate the provided fact, but a few weave in other information. 7.2% of paraphrases are either unrelated to the selected facts or paraphrase the fact incorrectly. Overall, 51.2% of messages have valid paraphrases.	45
3.4	Curiosity has 14,048 dialogs. On average, dialogs have 12.9 utterances. 60% of the assistants’ 90,534 utterances were liked.	46
3.5	The CHARM model outperforms end-to-end BERT on most tasks. We compare fact selection with MRR, dialog act prediction with micro-averaged F_1 , and like prediction with accuracy. Ablating dialog history degrades context-dependent tasks (fact selection and policy act prediction), but not tasks more dependent on one message.	49

3.6	✓ indicates a dataset has the feature, ⚠ that it does with a caveat, and ✗ that it does not. Conversational MS MARCO is a search dataset but has inquiry chains we want assistants to induce (exemplar in Appendix A.7). Topical Chat and Search as a Conversation are motivationally similar. While our dataset’s combination of (human) annotation is unique, all three datasets are steps forward in resources for conversational information-seeking.	51
5.1	The QANTA dataset is larger than most question answering datasets in QA pairs (120K). However, for most QB instances each sentence in a question can be considered a QA pair so the true size of the dataset is closer to 650K QA pairs. In Section 5.5 using sentence level QA pairs for training greatly improves model accuracy. The QANTA dataset has more tokens than all other QA datasets. Statistics for QANTA 2012 and 2013 only include publicly available data.	83
5.2	An entry from the gameplay dataset where the player correctly guesses “Atlanta” at word 47. The entry QID matches with the PROTO_ID field in the question dataset where additional information is stored such as the source tournament and year.	87
5.3	We assign each question in our dataset to either the train, development, or test fold. Questions in the development and test folds come from national championship tournaments which typically have the highest quality questions. The development and test folds are temporally separated from the train and development folds to avoid leakage. Questions in each fold are assigned a “guess” or “buzz” association depending on if they have gameplay data. Unassigned refers to questions for which we could not map their answer strings to Wikipedia titles or there did not exist an appropriate page to match to.	89
5.4	We compare several models by accuracy at start-of-question, end-of-question, and EW. In the table, models are sorted by start-of-question development set accuracy. Standard deviations for non-IR models are derived from five trials; standard deviation is not reported for the IR model since it is deterministic.	99
5.5	The table is an error breakdown for questions with at least twenty-five training examples. To analyze errors at the start of questions, we randomly sampled fifty errors and for end of question took all thirty-six errors. End of question errors are primarily wrong country errors as in Figure 5.16 where the model answers <u>United States</u> instead of <u>Spain</u> . Errors at the start of the question though are more diverse. The most common error is guessing the correct answer type, but not the specific member of that type; examples of this error class include answering <u>Albert Einstein</u> instead of <u>Alan Turing</u> , or <u>Iowa</u> instead of <u>Idaho</u>	104

5.6	The accuracy (ACC), expected wins (EW), and QB score (Score) of each buzzer on the validation set. Both MLP and RNN outperform the static threshold baseline by a large margin, but there is still a considerable gap from the optimal buzzer.	104
6.1	The topical diversity of the questions in the adversarially authored dataset based on a random sample of 100 questions.	120
6.2	The adversarially authored questions have similar n -gram overlap to the regular test questions. However, the overlap of the named entities (NE) decreases for IR Adversarial questions.	124
6.3	A breakdown of the phenomena in the adversarially authored dataset.	125
6.4	The first category of adversarially authored questions consists of examples that require reasoning. <u>Answer</u> displays the correct answer (all models were incorrect). For these examples, connecting the training and adversarially authored clues is simple for humans but difficult for models.	126
6.5	The second category of adversarial questions consists of clues that are present in the training data but are written in a distracting manner. <i>Training</i> shows relevant snippets from the training data. <i>Prediction</i> displays the RNN model’s answer prediction (always correct on Training, always incorrect on Adversarial).	127
A.1	Example dialog #1 from Curiosity. (U: User, A: Assistant)	150
A.2	Example dialog #2 from Curiosity. (U: User, A: Assistant). After mentioning the Green Party, the user asks a specific followup question; we use these interactions to estimate implicit preference.	151
A.3	A random sample of ten manually labeled paraphrases from the assistant. The top row indicates the label we (the authors) annotated, the middle row the message, and the bottom row the original fact from Wikipedia. The original fact is shown as displayed to crowd-workers including punctuation tokenization.	152
A.4	To compare like prediction between models, we randomly sample thirty dialogs and obtain predictions from CHARM and BERT. The table only shows messages where the model predictions disagree and indicates which model was correct. Dialogs are delineated by horizontal lines. Unfortunately, from only these examples we cannot determine why the CHARM model errors in most of these predictions.	153
A.5	An exemplar query chain from the conversational variant of MS MARCO. An ideal assistant should answer these questions <u>and</u> inspire these types of followup questions.	154
B.1	The linear model integrates a variety of features to determine which are most predictive of a subject responding correctly to an item. . . .	158

B.2	Table entries are Kendall’s τ rank correlation of IRT subject ability between rows and columns. Generally, the models agree on the ranking with the IRT-feas and IRT-disc having the strongest correlation.	160
B.3	Entries are Kendall’s rank correlation between rows and columns. Scores are SQuAD Exact Match (EM) and IRT-disc ability.	160
C.1	A random sample of QB answer strings and their matched Wikipedia pages. Answer mappings are easy to obtain accurately since most failures in exact matching are due to QB specific syntax that can be accounted for by rule based matching. Combined with manual annotation to find common non-exact matches, this process succeeds on 119,093 of 132,849.	164
C.2	A breakdown of NaturalQuestion example topics using QB categories. Most questions are about pop culture and the distribution includes many fewer questions about Literature and Fine Arts.	166
D.1	Failure and success cases for SCPN. The model fails to create a valid paraphrase of the sentence for 97% of questions.	167

List of Figures

1.1	An example of information-seeking dialog that the Curiosity dataset aims to support. Assistants should answer user questions <i>and</i> convey information that inspires meaningful followup questions.	2
1.2	QB questions are multi-sentence and reference uniquely identifiable answers (e.g., the <u>Apollo Program</u>). They begin with difficult clues and steadily progress to easy “giveaway” clues by the last sentence. In this question, the early clues are unknown to most while the final clue is known to most school-aged children. Here we show human gameplay data where players answered incorrectly (✗) and correctly (✓). Having above-average knowledge of this topic, my (correct) guess is shown by a ◇.	4
2.1	The Winograd Schema Challenge (Levesque et al., 2011) examples have two main ingredients. First, a statement containing two noun phrases and a pronoun which could refer to either (“trophy,” “suitcase,” and “it” in this example). Second, a single trigger word that when changed between two alternatives (“big” and “small”) changes which noun phrase the pronoun should resolve to. Well-crafted examples should expose models that do not “understand” text.	12
2.2	A sample question from TREC-8. Questions are contributed by participants or FAQFinder logs. Answers are text spans selected by NIST annotators from news articles.	15
2.3	In SQuAD (Rajpurkar et al., 2016), annotators read a passage from Wikipedia and write several questions. Answers to SQuAD questions are text spans such as “Colorado Desert.” Despite the QA in its name, SQuAD is better thought of as a RC dataset since questions are context-dependent.	15
2.4	Examples from NQ (Kwiatkowski et al., 2019) use real user queries from Google search and annotate them with long and short answers.	16

2.5	SQuAD’s questions are influenced by the distribution of Wikipedia pages, the set chosen to be part of SQuAD, and the text of the question. We call this generative process type <i>context-first</i> to emphasize that the context (paragraph in this case) directly influences the question.	18
2.6	In <i>question-first</i> QA, the question is created independently of any particular context (paragraph) or answer. This type of QA is most prevalent in information-seeking tasks like web search.	19
2.7	<i>Answer-first</i> questions are created by first thinking of a desired answer (or at least topic area), then crafting a question with that answer.	19
2.8	Deep averaging networks, recurrent neural networks (e.g., LSTMs and GRUs), and transformer networks (e.g., BERT) are common architectures in neural NLP models. To produce text representations of a fixed size (i.e., not a sequence of vectors), each architecture takes a different approach. Deep averaging networks average word vectors and then pass the subsequent vector through one or more feedforward layers; recurrent models often concatenate the last hidden state in each direction; transformer networks usually have a special token associated with the input’s general representation (in BERT, the CLS token).	26
2.9	The original QB interface and a popular modern interface for playing QB online. Both interfaces reveal questions word-by-word until a player interrupts the system, and makes a guess.	32
3.1	An example of information-seeking dialog that the Curiosity dataset aims to support. Assistants should answer user questions <u>and</u> convey information that inspires meaningful followup questions.	36
3.2	We sample pre-existing knowledge by asking users to indicate which <u>topically</u> related entities they already <u>know</u> . The assistant paraphrases facts related to either known entities (rooted facts), an aspect (aspect facts), or the topic generally (general facts). The user expresses engagement through a like button. Dialog acts are annotated in a separate crowd-source task.	37
3.3	In this example, the user is assigned to learn about <u>Lesotho</u> , specifically its <i>culture</i> and <i>history</i> . Given their topic, we users indicate which (related) entities—related to <u>Lesotho</u> —are familiar. Related entities range from relatively common like the <u>United States</u> to lesser known like <u>Basutoland</u> . We also provide guidelines and videos before crowd-workers start working on the task.	39
3.4	The user expresses the “interestingness” of the assistant’s messages through a “like” button (right of message). This is one of the two ways that we estimate user satisfaction.	40
3.5	The assistant can incorporate any number of facts into their reply to the user. Their goal is to answer the user’s immediate questions, and anticipate what information they would be most interested in.	41

3.6	User engagement is measured by dialog act followups (left) and like button usage (right). We compare reactions to messages that use a fact mentioning an entity the user knew about (rooted) and whether the fact is general or aspect-specific. Pairwise differences are statistically significant (99%+) with a two proportion z-test <u>except</u> for dialog act followups between rooted and non-rooted general facts. Overall, users prefer on-aspect, rooted facts.	43
3.7	Architecture: CHARM builds a dialog context up to $t = i - 1$ to predict the current message’s dialog acts (policy prediction) and the best facts to use. The model uses this combined with the current utterance to classify it’s dialog acts and if it will be liked.	46
4.1	Difficulty and Ability Discriminating (DAD) leaderboards infer the difficulty, discriminativeness, and feasibility of examples. Negative discriminability suggests an annotation error; for example, the question with most negative discriminability asks “Why did demand for rentals decrease?” when the answer is “demand for higher quality housing increased.”	55
4.2	A DAD leaderboard uses IRT to jointly infer item difficulty β_i , discriminability γ_i , feasibility λ_i , and subject skill θ_j . These predict the likelihood $p_{ij}(r_{ij} = 1)$ of a correct response r_{ij}	56
4.3	In IRT, the Item Characteristic Curve describes the probability that a specific item with difficulty β will be answered correctly as a function of skill θ . Visualizing the parameters is helpful in a few ways. First, it shows that high discriminability leads to larger differences in the maximal and minimal correctness probability (for a given skill range), and the tangent line visually links discriminability to the slope. Second, it clearly shows the maximal probability when feasibility is greater than zero. Lastly, it demonstrates that the inflection point is at the point where difficulty and skill equal out (in this case, zero).	57
4.4	We compare each IRT and linear model (LM) by how well they predict subject responses. We focus on ROC AUC since predicting responses is an imbalanced classification problem (most subjects are correct). Under that metric, all IRT models improve over the best LM, and the strongest LM ablation only uses IRT features. That textual features are predictive in the LM suggests they could improve future models.	61
4.5	Compared to the final ranking over a large test set, how well does a small test set correlate? The left shows correlation between mutually exclusive development set samples and the right between development samples and the full test set. In both experiments (panes), ranking systems by IRT ability is more stable—across all sample sizes—than mean accuracy and thus more reliable (Kendall’s rank correlation is higher). Bands show 95% confidence intervals of rank correlations across ten trials per sample size.	62

4.6	P-values of the rank correlation difference for each sample size and trial in Figure 4.5. The inherent noise in dev set sampling makes inferring significance difficult (left); test set driven results (right) are more significant.	64
4.7	Suppose we need to cold start, collect annotations for a new subject: what order would most rapidly increase correlation to the test data? As we expect, the correlations eventually converge, but with little data, IRT has better correlation than other methods. We suspect that the IRT information underperforms early on when the subject ability estimate is unstable.	65
4.8	This question is regarded as infeasible by the IRT model. Upon further inspection, the answer omits five acceptable answers, but more importantly does not permit all combinations of Turing machines.	66
4.9	We partition evaluation data by IRT difficulty and discriminability with accuracy in each quartile. Most improvements in high-accuracy systems come from getting high-difficulty questions right. items with low discriminability (and thus prone to annotation errors) are difficult for all subjects except the overfit ARGS-BERT model. We include top-performing SQuAD subjects, several notable subjects (systems), and a pair from the bottom of the leaderboard.	67
4.10	We annotate SQuAD items by discriminability, difficulty, and IRT prediction errors. For example, one question with negative discriminability was classified as “Wrong” with the explanation that the annotated answer indicates it is <i>not</i> answerable, but the question actually <i>is</i> answerable. items with negative discriminability or where IRT’s prediction is wrong have a much higher rate of annotation error (“Flawed” or “Wrong”). Using similar methodology, errors in datasets could be more rapidly identified.	68
5.1	QB is a trivia game where questions begin with clues that are initially difficult, but become progressively easier until a giveaway at the end of the question. Players answer as soon as they know the answer so as a result the earlier they answer the more knowledgeable they are. For example, answering after the first sentence indicates the player recognizes the librettist (Emanuel Schikaneder) and knows that they played Papageno in <i>The Magic Flute</i> (die Zauberflöte). Answering at the end of the question only requires surface knowledge of Mozart’s opera works.	72
5.2	Trivia has gone from a laid-back pastime to an organized, semi-professional competition format. The QB framework, in particular, which arose from College Bowl (US) and University Challenge (UK) emphasizes fairness and the ability to discover the better question answerer. As organizations such as the Academic Competition Federation and National Academic Quiz Tournaments emerged, the format has focused on academic, well-run tournaments.	74

5.3	Our interface and a popular modern interface for playing QB online. Both interfaces reveal questions word-by-word until a player interrupts the system and makes a guess.	82
5.4	Size of question answering datasets. Questions in the QANTA dataset have longer sentences than any other dataset. The instances from SimpleQuestions, SQuAD, and TriviaQA are comparatively short, which makes it less likely that they are as diverse as QB or Jeopardy!. For each dataset, we compare the lengths of questions rather than paired context paragraphs; to avoid the histogram being overly skewed we remove the top 5% of examples by length from each dataset.	84
5.5	Questions in QB cover most if not all academic topics taught in school such as history, literature, science, the fine arts, and social sciences. Even within a single category, questions cover a range of topics. Topically, the dataset is biased towards American and European topics in literature and history.	85
5.6	Distribution of wikidata.org answer types (“instance of” relation) further broken down by category. Most answers have matching types and reference a person, literary work, or geographic entity. Among these types, there is a good balance of answers spread across literature, history, fine arts, and science. Answer types with only one category are largely self-explanatory (e.g., mythological answers types to the mythology category). The special category “NOMATCH” are answers without a matched type and similar types are merged into larger categories.	86
5.7	Left: each protobowl user is represented by a dot, positioned by average accuracy and buzzing position; size and color indicate the number of questions answered by each user. Right: distribution of number of questions answered, accuracy, and buzzing position of all users. An average player buzzes with 65% of the question shown, and achieves about 60% accuracy.	88
5.8	The QANTA framework for playing Quiz Bowl with semi-independent guesser and buzzer models. After each word in the input is revealed the guesser model outputs its best guesses. The buzzer uses these in combination with positional and gameplay features to decide whether to take the buzz or wait action. The guesser is trained as a question answering system that provides guesses given the input text seen so far. Buzzers take on dual roles as calibrators of the guesser confidence scores and cost-sensitive decision classifiers by using the guesser’s score, positional features, and human gameplay data.	90
5.9	All our neural models feed their input to an embedding function, then a composition function, and finally a classification function. The primary variation across our models is the choice of composition function used to compute a fixed, example-level representation from its variable length input.	92

5.10	We plot the expected wins score with respect to buzzing position (solid dark blue). For the ten most played questions in the buz-ztest fold we show the empirical distribution for each individual ques-tion (dotted lines) and when aggregated together (solid light blue). Among the most played questions, expected wins over-rewards early buzzes, but appropriately rewards end-of-question buzzes.	98
5.11	The BERT and IR models are mostly wrong or correct on the same subset of questions. At the end of the question, most of the questions the BERT model is correct on, the IR model is also correct on.	100
5.12	A test question that was answered correctly by all models after the first sentence; a normally very difficult task for both humans and machines. A very similar training example allows all models to answer the question through trivial pattern matching.	101
5.13	Only the RNN model answers this question correctly. To test the ro-bustness of the model to semantically equivalent input modifications, we use SEARS-based (Ribeiro et al., 2018) synonym attacks and cause the model prediction to become incorrect. Although this exposes a flaw of the model, it is also likely that the low confidence score would likely lead a buzzer model to abstain; this highlights one benefit of implicitly incorporating confidence estimation into the evaluation.	101
5.14	The distribution of training examples per unique answer is heavily skewed. The most frequent answer (Japan) occurs about 100 times. Nearly half of the questions have one training example and just over sixty percent have either one or two training examples.	102
5.15	The more an answer is asked in the training set, the easier it is for all models, both at the start and end of the question. This is a significant source of errors since accuracy on at least 50% of test questions—those with seven or less training examples—is significantly lower for all models.	103
5.16	Although the answer to this question is <u>Spain</u> , many of the terms and phrases mentioned are correlated with the <u>United States</u> . Thus, the RNN model answers <u>United States</u> instead of the correct answer <u>Spain</u> . This is one of many examples where the model answers with the correct answer type (country), but incorrect member of that type.	105
5.17	In this question, the Threshold ▲ and MLP ■ buzzers are too aggressive and buzz before the guesser’s answer is correct. In contrast, the RNN ● is more conservative and buzzes shortly after the optimal point ◆ which is—by a wide margin—still earlier than the earliest (correct) human buzz ▼.	106

5.18	Comparing buzzers' behavior over time against the optimal buzzer. The red crossed area and dotted blue area combined indicates when the buzzer thinks that the guesser is correct, the other two combined when the buzzer thinks the guesser is wrong. The red (crossed) and orange (unhatched) areas combined indicates when the buzzer matches the optimal buzzer. Our goal is to maximize the red areas and minimize the blue areas. The static threshold baseline is overly aggressive, especially at earlier positions in the question (large dotted blue area); MLP and RNN both behaves reasonably well, and the aggressiveness of RNN is slightly more balanced early on in the question.	107
5.19	Breaking down the buzzer's performance on the individual question level. Impossible question means there is nothing the buzzer can do to beat the opponent. It is clearer that RNN performs better than MLP, making fewer mistakes of being overly aggressive.	108
5.20	The growth of the QANTA dataset in number of questions and number of distinct answers over the past twenty years starting in 1997. The dataset has grown by at least 5,000 questions every year since 2010. All questions with matched answers are included, and we construct the plot by using the tournament year of each question. Independently, participation in QB (and thus number of students writing questions) has roughly doubled every year since 2008.	112
6.1	Adversarial evaluation in NLP typically focuses on a specific phenomenon (e.g., word replacements) and then generates the corresponding examples (top). Consequently, adversarial examples are limited to the diversity of what the underlying generative model or perturbation rule can produce and also require downstream human evaluation to ensure validity. Our setup (bottom) instead has human-authored examples, using human-computer collaboration to craft adversarial examples with greater diversity.	115
6.2	This question is easily answered by a model after seeing the reference to "Un Bel Di." Our adversarial writing process highlights terms like these which humans can then modify to make clues more challenging for computer.	117
6.3	The author writes a question (top right), the QA system provides guesses (left), and explains why it makes those guesses (bottom right). The author can then adapt their question to "trick" the model.	118
6.4	The first round of adversarial writing attacks the IR model. Like regular test questions, adversarially authored questions begin with difficult clues that trick the model. However, the adversarial questions are significantly harder during the crucial middle third of the question.	122

6.5	The second round of adversarial writing attacks the IR and RNN models. The questions targeted against the IR system degrade the performance of all models. However, the reverse does not hold: the IR model is robust to the questions written to fool the RNN.	122
6.6	Humans find adversarially authored question about as difficult as normal questions: rusty weekend warriors (<i>Intermediate</i>), active players (<i>Expert</i>), or the best trivia players in the world (<i>National</i>).	123
6.7	The accuracy of the state-of-the-art Studio Ousia model degrades on the adversarially authored questions despite never being directly targeted. This verifies that our findings generalize beyond the RNN and IR models.	123
6.8	The interpretation successfully aids an attack against the IR system. The author removes the phrase containing the words “ellipse” and “parabola”, which are highlighted in the interface (shown in bold). In its place, they add a phrase which the model associates with the answer <u>Sphere</u>	127
6.9	The <i>Question Length</i> and the position where the model is first correct (<i>Buzzing Position</i> , lower is better) are shown as a question is written. In (1), the author makes a mistake by removing a sentence that makes the question easier for the IR model. In (2), the author uses the interpretation, replacing the highlighted word (shown in bold) “molecules” with “species” to trick the RNN model.	128
6.10	A failed attempt to trick the neural model. The author modifies the question multiple times, replacing words suggested by the interpretation, but is unable to break the system.	128
7.1	The ranking view and landing page of leaderboards should convey that the community values multiple types of progress. By highlighting the models according to different metrics, the ranking view de-emphasizes the importance of any single metric and encourages more thought into deciding what does the concept of “best model” mean. Lastly, rather than highlight only the highest-scoring models, we shift towards highlighting clusters of comparable models; for example, group all models whose scores are not statistically-speaking different.	140
7.2	In MRQA, TSNE shows a relationship between whether the task is NarrativeQA with respect to multidimensional difficulty and discriminability. The multidimensional IRT model uses six dimensions to match the six MRQA tasks.	143

A.1	The user is assigned two aspects about their topic. After they are satisfied with what they have learned about the first aspect, they click a button and switch to the next aspect. While the button click is not communicated to the assistant (the user must send a corresponding message), it resets the fact contextualizer; we observe that without this, too many facts were related to the previous aspect.	147
A.2	A short topic description is always visible to the assistant. The goal is to ensure the assistant always has a general understanding of the dialog topic.	147
A.3	To annotate dialog acts, we develop an interface that showed each utterance on a separate line. Annotators assign zero or more dialog acts to each utterance using grouped dropdowns.	148
B.1	The example from SQuAD with the lowest discriminability. Surprisingly, it had a <i>negative</i> discriminability, implying that the less skilled a subject is, the more likely their response is to be correct.	156
B.2	This example shows that the answer span is likely too large, causing models to fail in both SQuAD’s exact match and F1 metrics.	157
B.3	This highly discriminative question succeeds because there are many plausible answers. For example, although only “Turkish forces” is correct, some models answer “the Armenian state.”	158
B.4	The feasibility parameter λ of our IRT model represents the probability that an example is unsolvable. For example, annotation error could lead to an example always being scored incorrectly—regardless of how good the model is. In SQuAD 2.0, $\lambda < .434$ in the 5% percentile, $\lambda < .698$ for the 7.5%, and $\lambda < .931$ in the 10% percentile.	159
B.5	In SQuAD, TSNE shows a relationship between mean exact match (item accuracy) and answerability with respect to multidimensional difficulty and discriminability.	161

Chapter 1: Introduction

1.1 Overview

Asking questions—as in education and trivia games—is a powerful means of both learning and testing for understanding. The ability to ask questions combines aspects of intelligence unique to humans: language understanding, knowledge representation, and reasoning. Thus, building systems capable of intelligent question answering (QA) is a grand goal of natural language processing (NLP).

Humans primarily ask or answer questions for one of two purposes: to learn new information or to test the understanding of the answerer. These two purposes are the foundations of two (mostly) dichotomous, dual, and dueling perspectives in the evaluation of machine QA: the Cranfield paradigm (§2.1.1) and the Manchester paradigm (§2.1.2). In the Cranfield paradigm (Voorhees, 2002b), satisfying the user of a machine QA system is the central goal, regardless of how that is achieved. In contrast, the Manchester paradigm (Winograd, 1972; Levesque et al., 2011) emphasizes that evaluations should specifically test for *intelligent behavior* like its progenitor, the Turing Test (Turing, 1950). Although the choice of goal is rarely explicitly stated, that choice deeply shapes QA evaluations. This thesis begins by exploring how this choice influences evaluation and then turns to improving QA evaluations that test for *intelligent behavior*.

In QA, the dominant evaluation approach collects a large set of questions, pairs questions with correct answers, and automatically scores machine responses against reference answers (Voorhees, 2000b). The Cranfield experiments (Cleverdon, 1967) laid the foundation for modern information retrieval (IR) research by demonstrating that systems scoring higher on automatic evaluations corresponds to systems with higher user satisfaction. Consequently, the critical link in information-seeking scenarios is to ensure that reference answers satisfy users. This evaluation approach evolved to include comparisons to human effectiveness (Najberg, 2018) and was augmented by the construction of QA datasets with hundreds of thousands of questions.

When combined with machine learning and eventually deep learning, this led to feats like IBM Watson’s 2011 Jeopardy! victory over multi-time champions Brad Rutter and Ken Jennings (Ferrucci et al., 2010). However, an abundance of evidence has shown that while we can build “systems with very impressive performance, that [they] are nonetheless idiot-savants” (Levesque, 2014). In essence, the fact that one correctly answers “who discovered Hawking radiation?” does not necessarily imply intelligent QA.¹ The crucial oversight is that the same approach taken to evaluate

¹For example, answering with the person whose Wikipedia entry has the highest word overlap

U: <assistant wake-word>, tell me about Tahiti.
A: It's the largest island in French Polynesia, near the center of the Pacific
U: What is its history with France?

Figure 1.1: An example of information-seeking dialog that the Curiosity dataset aims to support. Assistants should answer user questions *and* convey information that inspires meaningful followup questions.

user satisfaction cannot be used to draw conclusions about intelligent behavior, because they have fundamentally different—although not unrelated—goals. Just as teachers ask questions to test student comprehension (Mehan, 1979), if our goal is to quantify progress towards intelligent QA we should create tests that better test that. A central idea throughout this thesis is improving discriminative power: the ability to distinguish between better and worse systems. We improve the discriminative power of QA benchmarks by introducing dynamic elements into their *scoring*, *format*, and *data*. Combined, our methods improve QA benchmarks so that they better measure progress towards intelligent QA.

1.2 Conversational Information-Seeking

We begin our exploration of the Cranfield paradigm in QA with perhaps the most natural form of information-seeking: conversation (Solomon, 1997). Information-seeking is naturally an interactive and iterative evolution of both the user’s information need and the expression of that need (Belkin et al., 1995). As partners in this information dance, machines should guide users toward resolving their information need (Radlinski and Craswell, 2017), even if they may not know precisely what they are looking for a priori (Raman et al., 2014). Towards the goal of satisfying information needs, the machine may be thought of as striving to be a clever and helpful librarian who converses with humans to develop and address information needs (Culpepper et al., 2018). Ultimately, the machine will be evaluated by whether in the course of its dialog, it met the user’s information need.

Conversational information-seeking sits at a convergence point between IR and NLP: IR researchers seek to make their systems more conversational while NLP researchers seek to make dialog systems more informative (Gopalakrishnan et al., 2019). Chapter 3 introduces a large resource for conversational information-seeking—an area that until recently lacked large resources despite great interest (Dalton et al., 2020). We introduce the Curiosity dataset (Figure 1.1) which has four features that combined make it unique: (1) it is designed for asymmetric conversational information-seeking (i.e., user–assistant interaction), (2) each dialog message is paired (possibly) with supporting documents, (3) each message has *explicit* user feedback, and (4) messages are annotated with dialog acts which are subsequently used to infer *implicit* user feedback. We validate both forms of user feedback by

to the question.

re-discovering a well-known principle in human learning: that novel information should be rooted in pre-existing knowledge (Chaiklin, 2003). In the spirit of the Cranfield paradigm, this work devises multiple ways to measure user satisfaction. Having seen one way to evaluate answers to questions, we shift to evaluation under the Manchester paradigm.

1.3 Questions are Not Equally Informative of Ability

A major goal of comparative evaluations is to reliably distinguish between better and worse models. In nearly all QA evaluations, since examples are weighted equally² they are implicitly assumed to be equally helpful in discriminating between models of disparate ability. However, this implicit foundational assumption is trivially falsifiable: datasets have erroneous questions that should not be weighed the same as good questions; even in a dataset free of errors, questions vary in their capability to discriminate between the knowledge of two players. For example, if the final sentence of Figure 1.2 were used to quiz world-class trivia players, it would do a poor job of discriminating between the players since the clue is too easy; in contrast, if only the first sentence was used, it would do a far better job. Similarly, if we tested novice players, then the situation would be reversed: the more difficult clue would likely not be answered by either player. Thus, the capability to infer skill is dependent on both the true skill of players *and* the intrinsic difficulty of questions.

Thus, questions are *not* equally informative, and we should change our evaluations to account for this. This precise problem was first encountered during the development of educational and psychological tests in the 1950s and 1960s (Traub, 1997). With respect to educational testing literature, current methods for QA evaluation (e.g., averaging scores) correspond to Classical Testing Theory (Edgeworth, 1888, CTT) which has largely been replaced by Item Response Theory (Lord et al., 1968, IRT) for the reasons discussed. Chapter 4 applies IRT to the *de facto* standard method for measuring progress in QA and many NLP tasks (Wang et al., 2019a)—the leaderboard. We show that jointly modeling the abilities of models and the difficulties of examples produces more reliable rankings by dynamically down-weighting the importance of less discriminative questions. Just as importantly, the joint model identifies poor test questions and efficiently guides annotation efforts which we incorporate in our discussion of future work (Chapter 7). Next, instead of increasing discriminative power through dynamic scoring, we do so with an intrinsically more discriminative QA format.

1.4 Incremental QA for Polytomous Evaluation of Knowledge

Trivia games are a popular means of testing knowledge and QA ability in humans and machines alike. This thesis considers two types of tests: dichotomous evaluations where answers are entirely correct or wrong, and polytomous evaluations where partial credit is given (e.g., one point out of five). Like most QA evaluations,

²Most evaluations average scores, which implicitly assigns equal weight to each individual score.

One part of this program included Stuart Roosa carrying redwood seeds. A part of this program failed in its mission to reach ✗ Fra Mauro, and the final part of this program brought geologist Harrison Schmitt ◊ to the Taurus-Littrow site. There was no second or third (*) mission in the commonly used numerical sequence for this program, which began with a mission that, in a test, killed Roger Chaffee, Ed White, and Gus Grissom in a ✗ fire. James Lovell was the hero ✓ of another failed mission in this program. Saturn ✓ V rockets were used to propel modules into space ✓ in, for 10 points, what ✓ NASA program whose eleventh mission landed Neil Armstrong on the moon?

Answer: Apollo Program

Figure 1.2: QB questions are multi-sentence and reference uniquely identifiable answers (e.g., the Apollo Program). They begin with difficult clues and steadily progress to easy “giveaway” clues by the last sentence. In this question, the early clues are unknown to most while the final clue is known to most school-aged children. Here we show human gameplay data where players answered incorrectly (✗) and correctly (✓). Having above-average knowledge of this topic, my (correct) guess is shown by a ◊.

the trivia game Jeopardy! dichotomously tests the knowledge of participants: only one player is awarded points, and players are only tested at a single point—the end of the question. As a consequence, when multiple players know the answer, Jeopardy! can be characterized as “kind of... a crappy video game where it’s, like, light goes on, press button—that’s it” (Malone, 2019). Thus, even if players have differing knowledge of question’s topic, they often have no way to display their expertise aside from impressively quick (jedi) reflexes (Harris, 2006).³ Putting the buzzer aside, the problem is that the only outcome which distinguishes between players of different ability is when one answers correctly and other answers incorrectly. In all other cases, we have no information about which player knows more. If we aim to distinguish between differing levels of QA ability, then dichotomous tests are an inefficient means of doing so. Chapter 5 addresses this oversight while simultaneously incorporating two mechanisms that bring it in line with the Manchester paradigm.

For this task, we turn to Quizbowl (QB)—a decades-old trivia game still played annually by over fifty thousand students across the world (National Academic Quiz Tournaments, 2020). In QB, each question tests knowledge incrementally by checking for knowledge of difficult clues first and easier clues last. This property of questions—called *pyramidal*—is implemented by creating multi-sentence questions where each subsequent sentence contains clues easier than previous ones (Figure 1.2). As in Jeopardy!, there is also a buzzer, but with one critical difference: it can be used at any point instead of only after the *full* question has been shown.

³In the Watson match, the reflexes of mere mortals could not match a computer’s, contributing to their demise.

In effect, QB is a polytomous evaluation⁴ with points deducted for every additional word needed to answer. This design combined with the pyramidity of QB questions makes it easier to distinguish between novices, dilettantes, weekend warriors, and GOATs.⁵

After introducing QB, Chapter 5 builds a framework that uses the trivia game in a Manchester-style QA evaluation. Specifically, we argue that evaluation for the purpose of evaluating intelligent behavior is improved by the pyramidity of questions, implicitly requiring models to “know what you don’t know”, and a metric that requires stability in guesses. To test this approach, we create a (deep-learning based) computer system for playing QB where one sub-system provides candidate answers while another decides when to buzz with that answer. We put this system (and our evaluation) to the test through a multi-year set of exhibition matches against accomplished trivia players; as in Jeopardy!, our system defeats even the most accomplished players despite using only statistical pattern matching. Upon closer inspection, although the *format* of QB is an improvement over prior QA evaluations, properties of the *data* still makes it easy for clever—yet cheap—statistical tricks to win the day.

1.5 Crafting Robust Questions Through Cooperative Machine-Human Authoring

Although answering a question to the satisfaction of a user may be sufficient in the Cranfield paradigm, if that question was answered through flawed reasoning then in the Manchester paradigm the test taker is not displaying intelligent behavior and should fail the test. For example, answering “who discovered Hawking radiation?” with Stephen Hawking on the basis of word co-occurrence statistics on Wikipedia is not sound; similarly answering Figure 1.2’s question on the basis that the answer to any question with the phrase “in this program” is the Apollo Program is also flawed and not how humans answer questions. The challenge then is “can [we] find questions where cheap tricks like this will not be sufficient to produce the desired behavior?” (Levesque, 2014). Levesque proposes three ways to make questions (more) robust to these tricks: make questions “Google-proof, . . . avoid questions with common patterns, . . . [and] watch for unintended bias” like word order betraying the answer. While following these recommendations is no panacea, following them increases the likelihood that a machine that correctly answers these questions is doing so through intelligent means.

In Chapter 6, we are the first to show that the same machines that used these tricks to best accomplished QB answers are *also* useful for helping humans to find and eliminate these shortcuts. At first, it may seem paradoxical that the “idiot-savant” models are useful for making questions more robust tests of intelligent QA. However, if we know—a priori—that a model explicitly uses pattern matching to

⁴Dichotomous responses are either correct or wrong; polytomous responses allow for partial credit.

⁵GOAT: the **G**reatest of **A**ll **T**ime.

answer, then we can use it to make it more difficult for pattern matching alone to derive the correct answer. Concretely, Chapter 6 designs an interface where humans and machines cooperatively and iteratively craft questions together: the machine identifies key phrases that are statistically predictive of the answer while the human creatively rephrases them. The output of this human–machine collaboration is a new set of questions that—by construction—are more difficult to answer through naive pattern matching *alone* and therefore if solved are more likely to indicate that the answer was derived through intelligent means.

1.6 Roadmap

In this thesis, we create methods that improve the format, data, and scoring frameworks in QA evaluations. Prior to discussing our methods, we first identify the Cranfield and Manchester paradigms as two influential perspectives in QA evaluations and show how they affect the form and data of evaluations (Chapter 2). In the subsequent chapter, as an example of the Cranfield paradigm, we introduce the Curiosity dataset along with a new idea for dialog QA evaluation (Chapter 3). From here, we turn to Manchester paradigm evaluation and begin by showing how lessons from educational testing can help us dynamically change the influence of individual examples (Chapter 4). Where the prior chapter held the format and data static, Chapter 5 changes the QA format to one that is intrinsically more discriminative. We use this same QA format in Chapter 6 to introduce a framework for cooperative human–computer question authorship that results in more robust evaluation data. Finally, Chapter 7 takes stock of the current state of QA evaluation and builds a broader vision that combines the ideas of this thesis.

Chapter 2: Background

We can only see a short distance ahead, but we can see plenty there that needs to be done.

Alan M. Turing, 1950

This chapter provides background in the epistemological heritage of question answering research (§2.1) and datasets (§2.3), and modern methods for question answering (§2.4). In discussing QA’s heritage, we more precisely define the origin and definition of the Cranfield and Manchester paradigms. Afterward, we compare several common QA formats. Using this new vocabulary, we analyze and categorize QA datasets by their purpose, how they were created, and other attributes which will help contextualize our dataset contributions in Chapters 3, 5, and 6. Following this, we describe modern (neural) methods for QA that we use throughout this thesis.

2.1 The Epistemological Heritage of Question Answering

Although machine QA is a comparatively young area of study, reviewing its history is helpful for understanding where the field has been, where it is going, and how to avoid repeating mistakes. In this section, we first compare the Cranfield paradigm (§2.1.1) and Manchester paradigm (§2.1.2) which are each motivated by related yet independent goals. The Cranfield paradigm is driven by the “goal of [information retrieval] research [which] is to improve access to information” (Croft, 2019). From this perspective, QA is another way to satisfy human information needs. The Manchester paradigm’s goal is to create QA systems that exhibit intelligent behavior. Neither is inherently better, but regardless the choice shapes the type of systems built. Far before the rise of big data, the exponential growth of computational resources, and deep learning, Schwitter et al. (2000) argue that:

There is general agreement that these competitive evaluations had a striking and beneficial effect on the performance of the various systems tested over the years. However, it is also recognized (albeit less generally) that these evaluation experiments also had the, [sic] less beneficial, [sic] effect that the participating systems focussed increasingly more narrowly on those few parameters that were measured in the evaluation, to the detriment of more general properties.

Arguably, this phenomenon has repeated itself (Dotan and Milli, 2020) with deep

learning, and it is only recently that other goals like robustness, fairness, or efficiency have been given attention (Paullada et al., 2020). Thus, we begin our review with the Cranfield paradigm, which paved the way for using large-scale static evaluations as a proxy for user satisfaction.

2.1.1 The Cranfield Paradigm

In information retrieval, the main task is to use a user’s stated information need (query) to select relevant information (documents) to show them. The most natural method to evaluate IR systems then would be to simply ask the user if the returned documents satisfy their information need. However, much like annotation in NLP this is expensive and time-consuming, so Cleverdon (1967) propose an alternative in the Cranfield experiments.¹ Rather than have users interact with systems, test collections are built, and all systems are evaluated on this collection. The collection has four parts: (1) a representative list of user queries, (2) a large set of documents, (3) per-query relevance judgments indicating which documents were relevant to each query, and (4) a means to aggregate per-query scores. Instead of putting humans in the loop with IR systems, “in the Cranfield paradigm, researchers perform experiments on test collections to compare the relative effectiveness of different retrieval approaches” (Voorhees, 2002b).

The Cranfield paradigm makes three key assumptions: the relevance of one document is independent of all the others, a single set of judgments is representative of the user population, and all relevant documents are annotated (Voorhees, 2019). At first glance, it would seem to be that every assumption is violated with real users: they do not like repeated information, users may differ in judging relevance, and it is infeasible to annotate every query-document pair unless the document collection is small (Cleverdon, 1991). Indeed, in the time of the Cranfield experiments these arguments made test collections seem like a radical proposal (Taube, 1965).² Despite this, part of the success of the Cranfield paradigm and subsequently TREC’s success is attributable to great care in aligning the core task with the evaluation (Jones, 2001); for example, repeated information does not matter if the user only requires *at least* one relevant document.

As IR systems became more effective at retrieving documents to sort queries, researchers started working on extensions that returned short answers instead of whole documents (Sanderson and Croft, 2012). Despite the change in output format, the core goal mirrors that of IR: return relevant information—in the form of a short answer—to the user. For example, in the first large-scale evaluation of domain-independent QA systems (TREC-8) “human assessors read each string and made a binary decision as to whether the string actually did contain an answer to the question” (Voorhees, 2000b). With this history lesson in mind, it should be clear by now that although this is a reasonable means to evaluate how effective a

¹The Cranfield experiments and thus the Cranfield paradigm are named after where they were conducted: Cranfield University in Cranfield, Bedfordshire, United Kingdom.

²This was in part due to its controversial but ultimately correct finding that word-based indexing of documents was effective.

QA system is for *users*, there is (rightfully) no focus on evaluating whether the system is behaving intelligently. Despite this, system effectiveness on QA benchmarks have been repeatedly used to claim super-human—and presumably “intelligent”—capabilities (Najberg, 2018). Next, we formalize an alternate paradigm that better aligns with the goal of testing for intelligent behavior than the Cranfield paradigm.

2.1.2 The Manchester Paradigm

The Manchester paradigm is an intellectual descendant of the Turing Test (Turing, 1950). Just as the Cranfield paradigm is named for Cranfield University, we name the Manchester paradigm after where Alan Turing created the Turing Test: the University of Manchester. Rather than consider the ill-defined question of “can machines think?” Turing proposed that we instead consider the imitation game: a test of whether a machine could fool an interrogator into thinking it was a human.

The ELIZA (Weizenbaum, 1966) dialog system is the first well known, QA-like system to have “passed” a weak form of the Turing Test; related systems like PARRY (Colby, 1981) eventually followed. ELIZA was intentionally built to “dazzle even the most experienced observer” while still being a “mere collection of procedures.”³ Consequently, the lesson from ELIZA is that “we often read into a program’s behavior our own ideas of what it understands” (Winograd, 1977) which is sometimes referred to naive psychology (Watt, 1996); Caporael (1986) also point out this anthropomorphization of machines. In the paper describing ELIZA, Weizenbaum makes the point that there are limits to ELIZA’s “understanding,” since a crucial component of understanding “is not the subject’s ability to continue a conversation, but to draw valid conclusions from what is being told.”

We posit that Terry Winograd’s SHRDLU QA system (Winograd, 1971) is a response to this criticism. Winograd viewed SHRDLU as a “system [that] attempts to integrate all the aspects of language, in combining linguistic knowledge with a command of the world being discussed” (Winograd, 1977). This system builds an internal model of a toy block world that is manipulated through natural language; in this way, SHRDLU draws valid conclusions about the toy block world. In a similar spirit, a core goal of the Manchester paradigm is to test—through a natural language interface—if QA systems can manipulate a model of the real world to derive correct answers.

There is, however, a crucial difference in the feasibility of testing for user satisfaction in the Cranfield paradigm and testing for intelligent behavior in the Manchester paradigm. Namely, the main challenge that the Cranfield paradigm addresses is one of cost: it is simply not *practical* to evaluate every new system with humans. In contrast, the Manchester paradigm must contend with the reason for the Turing Test in the first place: that we do not know how to define whether

³Other early QA systems included BASEBALL which manipulated a knowledge based of football games through natural language (Green et al., 1961) which inspired later work in querying databases (Copestake and Jones, 1990), LUNAR which helped Apollo Program scientists find specific Moon samples matching certain traits (Woods, 1972), and STUDENT that solved simple word algebra problems by parsing numerical expressions (Bobrow, 1964).

a system is intelligent, in large part because defining intelligence is at minimum exceptionally challenging.

Computer scientists like Turing were not the first to struggle with defining, understanding, and measuring intelligence: this is a long-standing challenge in fields like psychology. While the nature of intelligence is highly contested, prominent theories include the g factor theory which postulates that there is a single underlying general intelligence (Spearman, 1904), theories that postulate multiple independent axes of intelligence such as primary mental abilities (Thurstone, 1973) or multiple intelligences (Gardner, 2011, p. 277),⁴ and a triarchic theory based on the idea that human intelligence is a combination of componential, experiential and practical components such as “mental activity directed toward purposive adaptation to, and selection and shaping of, real-world environments relevant to one’s life” (Sternberg, 1985). Of these, the g factor theory took strongest hold and was arguably central in the creation of psychometrics: a field studying the measurement of intelligence in humans (Section 2.6).

While we will not opine as to which theory is correct,⁵ it is not necessary to do so to discuss why Sternberg’s triarchic theory best lights the way when creating evaluations of machine intelligence.⁶ At its core, the g factor theory posits that manifestations of intelligence—such as demonstrating mathematical or spatial reasoning—emerge from a general mental ability (the g factor). The validity of measuring the g factor (i.e., IQ) in humans is questionable (Section 2.6),⁷ and makes even less sense for machines: every system discussed thus far—by construction (e.g., ELIZA)—does not have an underlying “intelligent” core. While modern statistical learning methods have certainly pushed state-of-the-art, they do little to change this state of affairs (Bender and Koller, 2020). Thus, at best, approaching the evaluation of machine intelligence from this perspective implies measuring a quantity (g factor) that we know does not exist in machines built with current technology.

Another family of intelligence theories—such as primary mental abilities or multiple intelligences—posit that intelligence is the combination of several abilities such as numerical, reasoning, musical, and bodily-kinesthetic ability. While these abilities map onto machines with varying degrees of success (e.g., bodily-kinesthetic ability is irrelevant to most QA systems), these theories are nonetheless helpful for creating a taxonomy of general skills that we should endeavor to make machines capable of. For example, developing machines capable of numerical ability and testing for that ability is tractable and an example of what we have referred to in this thesis as an intelligent behavior. However, theories of multiple intelligences still leave much on the table. These theories take a narrower skill-based view that omits

⁴Gardner’s multiple intelligences can be understood as multiple independent abilities like linguistic or numerical reasoning.

⁵This is an empirical question which we do not investigate in this thesis.

⁶Furthermore, the nature of human intelligence may be different from artificial general intelligence (provided of course that AGI is achievable).

⁷Although multiple studies have demonstrated correlation of IQ test results with positive outcomes like job success, as the old adage goes, correlation is not necessarily causation.

several aspects of intelligence included in the triarchic theory such as creativity (e.g., composing novel questions as in Chapter 6), generalization from experience (e.g., few-shot learning as in Section 5.10.1), adapting and influencing environments (e.g., reinforcement learning), and the ability to learn efficiently. While developing machines capable of achieving particular skills—such as reasoning from known facts—is of definite interest in NLP, this broader view covers these skills⁸ while describing additional aspects of intelligence that are already of active scientific interest in NLP, machine learning, and AI.

In this spirit, an alternative interpretation of the Turing Test is that it is “not asking whether all digital computers would do well in the game nor whether the computers at present available would do well, but whether there are imaginable computers which would do well” (Turing, 1950). One way to build towards this goal is to iteratively imagine tasks that a machine *should* be able to do at least as well as a human, prototyping a machine ostensibly capable of that task, and then testing for that capability. Likewise, we can iteratively imagine aspects of intelligence—such as sample-efficient learning or generalization to novel contexts—that humans regularly exhibit, develop ways to build those in machines, and measure our progress. In this thesis, we refer to both of these manifestations of intelligence as intelligent behaviors. Towards the goal of building machines exhibiting increasingly many intelligent behaviors, descriptive theories and definitions of intelligence are helpful for identifying intelligent behaviors that we can work towards developing in machines.

In testing for capabilities, we introduce the second element of Manchester paradigm: we seek not to prove that a machine has a capability representative of a specific intelligent behavior, but to know when it does not so we can work towards building that capability. Through this mechanism, the Turing Test “represents what it is that AI must endeavor eventually to accomplish scientifically” (Harnad, 1992). For example, one intelligent behavior humans exhibition is being able to answer a question regardless of how it is phrased (Chapter 6). The challenge however comes in testing for this: we cannot tractably test all possible rephrasings. Theory also suggests this intractability: if the imitator model and interrogator are Turing Machines (Turing, 1937), then determining whether the imitator is a machine is undecidable (Sato and Ikegami, 2004). Thus, we avoid positively *proving* ability or passing the Turing Test since it is likely impossible. A wider view of this point partially explains the “moving goalposts” phenomenon in AI research where once a grand challenge is solved—like Chess (hsiung Hsu et al., 1995) or Go (Silver et al., 2016)—it is no longer considered “AI.” Instead, under the Manchester paradigm, the process of creating tests, improving tests, and improving models based on test failures guides research.

The best and most famous example of the Manchester paradigm is the Winograd schema challenge (Levesque et al., 2011). In this QA task, machines answer specially crafted questions with binary answers (Figure 2.1). Examples in the Winograd schema challenge always mention two parties (noun phrases), one is referred

⁸These academically-oriented skills reside in the componential analytical subtheory.

Questions	Answer
The trophy would not fit in the brown suitcase because it was too <u>big</u> .	
What was too <u>big</u> ?	trophy /suitcase
The trophy would not fit in the brown suitcase because it was too <u>small</u> .	
What was too <u>small</u> ?	trophy/ suitcase

Figure 2.1: The Winograd Schema Challenge (Levesque et al., 2011) examples have two main ingredients. First, a statement containing two noun phrases and a pronoun which could refer to either (“trophy,” “suitcase,” and “it” in this example). Second, a single trigger word that when changed between two alternatives (“big” and “small”) changes which noun phrase the pronoun should resolve to. Well-crafted examples should expose models that do not “understand” text.

to by a pronoun or possessive adjective, the question involves determining the referred noun party, and crucially there is a special word that when replaced by an alternative, flips the answer. Should a machine fail to answer questions like this, we conclude it is not displaying intelligent behavior and is thus not intelligent in at least the same way humans are. The opposite is not necessarily true though, since we only test for a single case and would further need to make the logical inference that the Turing Test tries to avoid: to define intelligence itself as displaying a specific list of intelligent behaviors. The key feature of the Winograd schema challenge, its successor the Winogrande challenge (Sakaguchi et al., 2020), and Manchester paradigm QA evaluation is to identify specific intelligent behaviors that the evaluation should test for.

This distinction of paradigms is a generalization of a point made by Gardner et al. (2020b)—that the three broad motivations for QA are to (1) fill human information needs, (2) probe a system’s understanding of some context (Weston et al., 2016), and (3) to transfer learned parameters.⁹ Specifically, filling human information needs aligns with the Cranfield paradigm and the Manchester paradigm generalizes the idea of probing a system’s understanding of a context to the idea of testing for intelligent behavior which may not necessarily require explicit context at all.¹⁰ While we have introduced these as opposing paradigms by contrasting their main goals, there is a wide swath of research that advances the goals of both.

2.1.3 Common Ground

Although we present these paradigms as a dichotomy, the world is not black and white, and progress is often shared through common sub-goals. Making models

⁹Parameter transfer is based on the idea that if the end task follows a QA format, that training on QA-like data may be helpful even if the task itself is not QA.

¹⁰For example, answering many semantically equivalent phrasings of the same question.

more robust is—to an extent—one of these areas of overlap. For example, Microsoft Research and Bing created the Web Scale Speller Challenge in 2011 (Wang and Pedersen, 2011) to motivate research in making search engines more robust through better handling of spelling errors. In concurrence with the Cranfield paradigm, making search engines robust to mis-spellings ultimately helps users find information faster. From the Manchester paradigm perspective, humans demonstrate an impressive robustness to poor spelling, albeit at speed penalty (Rayner et al., 2006),¹¹ so expecting this of machines is quite reasonable. Along similar lines, Contrast Sets (Gardner et al., 2020a) test robustness to syntactic variants of the original questions in multiple datasets. However, there is a limit to this: when robustness works towards the goals of one paradigm, but against the other.

Suppose a user wanted to find a particular email, but only could recall a mentioned word that is otherwise unrelated to the email’s contents. In the abstract, what we really ask is whether we should allow a system to use information that although helpful in solving the task, is not related to any conventional sense of natural language understanding or reasoning. The Cranfield paradigm would certainly welcome this information; in contrast, the Manchester paradigm would reject this usage as one of a larger class of correlations that models should not rely on (Feng et al., 2018). Beyond robustness, these paradigms share interests in areas like few-shot and zero-shot QA: responding to infrequently asked questions is important in practice (Baeza-Yates et al., 2007), and building models that learn more from less certainly qualifies as intelligent behavior (Linzen, 2020). Next, we introduce common QA formats and use this as a means to survey of QA and reading comprehension (RC) datasets.

2.2 Question Answering Dataset Characteristics

In addition to a QA dataset’s goals, they are also characterized by the QA format (§2.2.1), how questions are created (§2.2.3), and the context that they assume (§2.2.2). Questions based on specific text passages (e.g., from Wikipedia) will be different from questions that you might ask a digital assistant or search engine; likewise, the form of the answer also influences the data (e.g., multiple-choice versus free response). Ultimately, each of these factors changes the data distribution and the types of models built for the task.

¹¹Davis (2003) summarizes human robustness by saying that: *Aocdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mtttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit plcae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihis is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.*

2.2.1 Answer Formats

All QA formats share three components: the question, an implicit or explicit context, and a desired answer.¹² We begin by first looking at the forms that answers take.

Although it may seem easy to check the correctness of answers, this is a major challenge in QA and practitioners frequently realize upon inspecting the data that “there is no such thing as a question with an obvious answer.” (Voorhees and Tice, 2000). Along similar lines, the creators of NaturalQuestions—a QA dataset derived from Google search queries—estimate that one third of (short) answers to questions are “debatable” (Kwiatkowski et al., 2019).¹³ In the early days of large-scale QA evaluation with TREC (Voorhees, 2000b), human annotators graded each system’s answers, but this was ultimately not feasible as the numbers of questions and systems grew. Much like in IR, QA moved to (mostly) automatic methods to evaluate answers from QA systems.

Free Response Although free response is conceptually the simplest and relatively straightforward for humans to evaluate, it is the most challenging format to automatically grade. For example, is “Turing” a good enough answer to “who proposed the imitation game?” compared to “Alan Mathison Turing?” Would this be changed if instead of a uniquely identifying name like Turing, the answer were a much more common name like “Pedro Rodriguez” which could refer to no less than nine famous athletes, several in the same sports? One approach taken by the Bing-based QA dataset MS-MARCO (Nguyen et al., 2016) is to compute textual overlap between reference answers and proposed answers with BLEU-1 (Papineni et al., 2002) and ROUGE-L (Lin, 2004). Unfortunately, these metrics often have unimpressive correlation to human judgements (Callison-Burch et al., 2006; Chen et al., 2019) as in SemEval-2018 Task 11 (Ostermann et al., 2018), especially when single word differences have significant effects on correctness (Yang et al., 2018a).¹⁴ While evaluating free response answers remains challenging, the expressiveness of the format continues to encourage the improvement of scoring metrics through the use of learned metrics (Nema and Khapra, 2018; Chen et al., 2020).

Span Selection One step back from free-response, is answering questions using text spans from documents. The first large-scale examples of this evaluation format are TREC-QA-8 (Voorhees, 2000b) and TREC-QA-9 (Voorhees, 2000a). In this task (Figure 2.2),¹⁵ systems answer questions with spans of text from a document in a

¹²In this case, “answer” is more accurately described as a desired response, whether it be multiple answers or abstention.

¹³They define correct (but debatable) as: “a reasonable person could be satisfied by the answer; however, a reasonable person could raise a reasonable doubt about the answer.”

¹⁴Answers like “the radiation of wireless routers has **[an/no]** impact on people” have high lexical overlap despite being contradictory.

¹⁵As of writing, we have not lost contact, although that is expected in the 2020s or early 2030s.

Question: When will the Voyagers lose contact with Earth?
Answer: about 2015 or 2020

Figure 2.2: A sample question from TREC-8. Questions are contributed by participants or FAQFinder logs. Answers are text spans selected by NIST annotators from news articles.

Page: Southern California
Passage: To the east is the **Colorado Desert** and the **Colorado River** at the border with Arizona, and the Mojave Desert at the border with the state of Nevada. To the south is the Mexico–United States border.
Question 1: What is the name of the water body that is found to the east?
Answer 1: **Colorado River**
Question 2: What is the name of the desert on the border of Arizona?
Answer 2: **Colorado Desert**

Figure 2.3: In SQuAD (Rajpurkar et al., 2016), annotators read a passage from Wikipedia and write several questions. Answers to SQuAD questions are text spans such as “Colorado Desert.” Despite the QA in its name, SQuAD is better thought of as a RC dataset since questions are context-dependent.

pre-specified collection.¹⁶ This format was later adopted by both versions of the Stanford QA dataset (Rajpurkar et al., 2016, 2018, SQuAD) (Figure 2.3) with the second iteration incorporating abstention as in TREC 2001 (Voorhees, 2001). The format has proved popular and is also used by NaturalQuestions (Figure 2.4) and many others. Beyond having some of the lexical matching problems as free response answers, the primary disadvantage of span selection is that QA models are limited to using only text present in the provided documents.

Multiclass Classification Although free-response and span selection are flexible, one drawback is that sometimes it is unclear which semantic concept is intended (e.g., Francis Bacon the philosopher versus the artist). In human trivia games like Jeopardy! or Quizbowl (Chapter 5), this is easily solved by prompting humans to be more specific. The equivalent for machines is to answer from a large, but closed set of answers, each of which refer to a distinct concept such as named entities. By construction, answers to examples in WebQuestions (Berant et al., 2013) and SimpleQuestions (Bordes et al., 2015) are named entities from knowledge bases like Freebase (Bast et al., 2014). Classification is also the format we use in QB evaluation since most answers correspond to Wikipedia page entities (§5.3.4). The primary tradeoff of this evaluation scheme is that although answers are never ambiguous, if the set of candidate answers does not encompass all possible answers, then the

¹⁶In TREC-8, since the requirement that the document support the span was not originally stated, NIST allowed these “unsupported” answers, but adopted this requirement for TREC-9.

Page:	<u>Sun</u>
Question:	what stage of the star life cycle is the sun in
Long Answer:	The Sun is about halfway through its main-sequence stage , during which nuclear fusion reactions in its core fuse hydrogen into helium. Each second, more than four million tonnes of matter . . .
Short Answer:	about halfway through its main-sequence stage

Figure 2.4: Examples from NQ (Kwiatkowski et al., 2019) use real user queries from Google search and annotate them with long and short answers.

evaluation is less representative of true model effectiveness.

Multiple Choice The final common answer format is multiple-choice, much like one might find on standardized education tests. This format is attractive from the modeling perspective since answering from a small set of options is a more tractable challenge. Along these lines, there are multiple-choice tasks for story (Richardson et al., 2013, MCTest), science (Welbl et al., 2017; Mihaylov et al., 2018; Clark et al., 2018), and commonsense reasoning (Talmor et al., 2019) questions. The main drawback to multiple-choice though is that creating plausible false answers is challenging and requires substantial conscious effort (Welbl et al., 2017; Talmor et al., 2019) or frameworks like the Winograd schema challenge or templates of understanding (Dunietz et al., 2020).

Abstention Another common feature of tasks since TREC-QA 2001 (Voorhees, 2001) and QA4MRE at CLEF (Peñas et al., 2013) is the possibility of a question having no answer and penalties for wrong answers. TrecQA 2002 took this a step farther and factored in the confidence scores instead of a simple binary decision (Voorhees, 2002a). The motivation behind this is that models should know what they do not know (Rajpurkar et al., 2018); Quizbowl takes this even farther by requiring this decision to be made many times per question, instead of only once. In other terms, models should have calibrated estimates of their confidence (Kamath et al., 2020) to help decide when there is legitimately no correct answer or when the model’s best guess is wrong.

Having covered the most common forms that answers take,¹⁷ we next discuss the information contexts used in various datasets before bringing everything together to characterize common generative stories of QA datasets.

2.2.2 Information Context

We—as humans—draw on our knowledge and (sometimes) external sources (e.g., webpages or books) to answer questions. Although these sources are varied, when we evaluate QA models they are limited to the information in the training

¹⁷Although extensive, our list does not cover formats like lists of answers.

data or passed in as input. Generally, the format of QA examples either assumes a global context and passes only the question to the model, or it provides a specific context—such as a paragraph—for the model to use.¹⁸

Global Context Examples of global context QA include tasks like Jeopardy! (Dunn et al., 2017), Quizbowl, and ComplexWebQuestions (Talmor and Berant, 2018). In these tasks, there is no provided context, and any knowledge can answer the question. Thus, tasks based on search queries like early iterations of TREC-QA, MS-MARCO, and NaturalQuestions are all global context questions.¹⁹ Although factoid QA is the most common type, some common sense reasoning tasks also fit this description.

Local Context In contrast, other tasks require that models use a specific, local context to answer the question. The most obvious examples of this are story-based datasets that ask questions about what occurred in the story (context) to test reading comprehension (Kočíský et al., 2018; Tafjord et al., 2019). This is also common in factoid QA tasks like SQuAD or DROP (Dua et al., 2019) where (frequently) the question requires a paragraph (context) for it to be answered unambiguously (e.g., in Figure 2.3). In either case, the differentiator is not whether a certain local context is helpful in answering a question, but if it—in general—is required.

Another example of local context are conversational QA datasets which intentionally make the dialog history important to answering questions. The earliest large sequential QA dataset is TrecQA 2004 (Voorhees, 2004), but it has been followed up on by many other likes CoQA (Reddy et al., 2019), QuAC (Choi et al., 2018),²⁰ and the Curiosity dataset (Chapter 3).

2.2.3 Generative Story

Finally, having discussed the format and context of questions, we are prepared to treat them as part of the generative process in creating a question. If we view the creation of questions as sampling from a probability distribution, then specifying its generative process is helpful in identifying latent factors and observed factors that influence the types of questions asked. Although there are several latent factors we could define, such as the difficulty of questions as we explore in Chapter 4, for now we focus on observed factors. In particular, we model the data distribution of questions as having three factors: the question, the context, and the answer.

Under this generative model, the defining aspect of the question distribution is the choice of which factor is selected or sampled first. For example, if we select a context paragraph first, then the question will be influenced and perhaps

¹⁸Datasets also typically assume a particular temporal context (i.e., when it was created); an assumption tested in TREC-QA 2006 (Dang et al., 2006) with questions like “who is the president of the U.S.” whose answer is temporally dependent.

¹⁹Although NaturalQuestions provides gold passages with answers, questions are created independently of the document.

²⁰Elgohary et al. (2019) create a de-contextualized version of QUAC called CANARD.

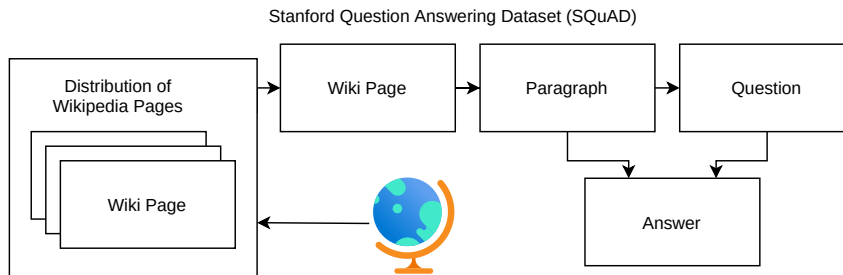


Figure 2.5: SQuAD’s questions are influenced by the distribution of Wikipedia pages, the set chosen to be part of SQuAD, and the text of the question. We call this generative process type *context-first* to emphasize that the context (paragraph in this case) directly influences the question.

explicitly depend on that context; on the other hand, if the question is created first (e.g., a search engine question) then it does not depend on any particular context. Although the Cranfield and Manchester paradigms are a separate concept, identifying the generative story is often helpful in identifying the core motivation of a QA dataset. When the answer to a question is chosen first—such as a teacher writing a test question—the task almost surely falls under the Manchester paradigm since by construction it does not originate from a genuine information need. Next, we describe these three generative stories for questions.

Context-First QA *Context-first* questions are created when (optionally) a topic is selected, then a paragraph discussing the topic, and then a question is written that uses the context. In every step along the way, there are multiple points of influence—none of which are necessarily good or bad. As an example, let’s consider SQuAD 1.0 (Rajpurkar et al., 2016) where questions are written based on paragraphs from one of 490 Wikipedia pages (Figure 2.5).²¹ Cultural artifacts like Wikipedia—which many QA datasets such as SQuAD depend on—are reflective of society and carry its biases (Reagle and Rhue, 2011); in Wikidata, which is partially derived from Wikipedia, 77% of people entities are male which in turn influences who is asked about.²² Similarly, while it is obvious that the question depends on the context, a somewhat less predictable side effect is that questions are often lexically similar to the context which makes them easier to answer through pattern matching (Sugawara et al., 2018; Rondeau and Hazen, 2018). This exact effect was observed years earlier in TrecQA-8²³ where questions “were often back-formulations of statements in the documents, which made the questions somewhat unnatural and also made the task easier since the target document contained most of the question words” (Voorhees, 2000a). In response, the next year’s track used search queries from Microsoft’s Encarta and Excite instead which induces a different generative story. Aside from

²¹442 in the training set and 48 in the development set.

²²We used the Wikidata query interface to derive this number.

²³Track participants submitted their own questions and were evaluated on questions from other participants.

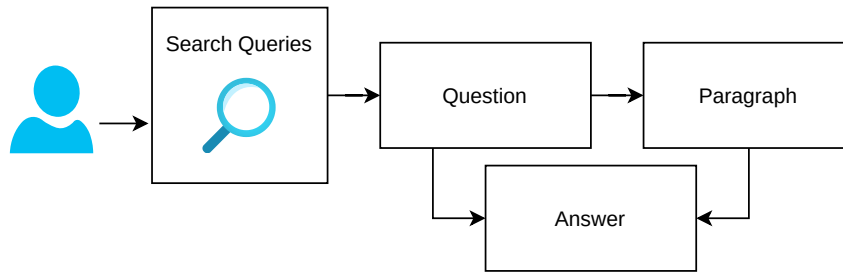


Figure 2.6: In *question-first* QA, the question is created independently of any particular context (paragraph) or answer. This type of QA is most prevalent in information-seeking tasks like web search.

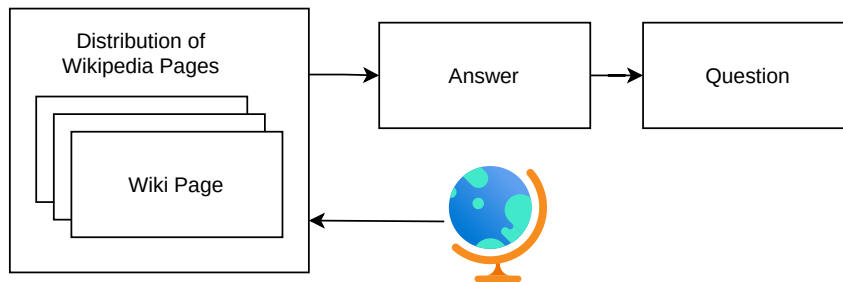


Figure 2.7: *Answer-first* questions are created by first thinking of a desired answer (or at least topic area), then crafting a question with that answer.

factoid QA, context-first QA is common in story or narrative QA questions that frequently tests if models make specific inferences from the story.²⁴

Question-First QA In this generative process, the question is the first component created—datasets based on search query logs are an excellent example of this process. In question-first QA, questions often emerge from an information need and therefore presumably the answer—if there is one—is not known, so does not directly influence the question. This would naturally include many of the TrecQA tasks, MS MARCO, and NaturalQuestions. While some of these tasks provide supporting documents, their inclusion does not directly influence the questions themselves since even questions with no answer are included. In these tasks, although providing a question-derived context is common, it is not a necessary component.

²⁴Consider an example from [Dunietz et al. \(2020\)](#): Why did the dog come inside? Because: (1) it was raining outside, (2) dogs prefer not getting wet to getting wet, and (3) going outside while it is raining would get you wet as opposed to going outside when it is not raining and being dry. They propose to continue this questioning until the resulting facts are obviously true.

Answer-First QA The final major category of generative stories is *answer-first* QA where the answer is the first component conceived (Figure 2.7). In contrast with question-first QA, the answer or at least the topic of the question is specified a priori. This is most prevalent in trivia tasks like Jeopardy!, Quizbowl, and TriviaQA (Josh et al., 2017) where the question writer aims to test a subject’s understanding or knowledge of a topic. One drawback of this approach—much as with context-first—is that often the distribution of answers is heavily skewed (§5.7.2). Like question-first, although datasets like TriviaQA also provide supporting documents, they again do not directly influence the question like a priori specification of the answer does. All this said, if the goal is to test knowledge of a particular answer or topic, answer-first is the go-to format.

Human and Machine Authorship Although some datasets are entirely human-written, machines increasingly play a larger role in dataset construction. Whether a dataset is human-authored (👤/🧑), machine-authored (🤖), or authored by a combination *also* affects the distribution of questions—Chapter 6 investigates how cooperative human-machine authoring affects question difficulty. For example, bAbI is generated from a set of templates and thus is entirely machine-authored (Weston et al., 2016), WikiHop from a knowledge base by posing slot-filling as QA (Welbl et al., 2018), and WikiReading from aligning WikiData properties to contexts automatically (Hewlett et al., 2016). While this automated approach makes it easy to quickly generate many questions, they tend not to be as diverse as human-authored examples. Still, these approaches show promise in testing for specific capabilities and comparing to more naturally constructed datasets (Liu et al., 2021). SimpleQuestions (Bordes et al., 2015) and Complex WebQuestions (Talmor and Berant, 2018) take a middle-of-the-road approach and automatically select what relationships a question should ask about and then human annotators phrase them (🤖→👤/🧑). Under this setup, an annotator might receive a knowledge graph triple like (Picard, played by, Patrick Stewart) and write “who played Picard, the captain of the Enterprise?” In another model for human-machine authorship, humans write questions and machines filter out questions that are too easy (👤/🧑→🤖) as in DROP (Dua et al., 2019) and QUOREF (Dasigi et al., 2019). Chapter 6 takes this a step farther and interactively constructs questions (🤖↔👤/🧑) like subsequently done in dialog (Nie et al., 2020), natural language inference (Nie et al., 2020), and the Winograd challenge (Sakaguchi et al., 2020).









On the other end are questions authored entirely by humans, be it through crowdsourcing (🧑) or some other means (👤). This distinguishes tasks like Quizbowl or Quasar-T (Dhingra et al., 2017) which are created by domain experts from crowdsourced tasks like SQuAD, CoQA (Reddy et al., 2019), and QuAC (Choi et al., 2018). In the case of trivia games, writers have intrinsic motivations for crafting good questions, players are intrinsically motivated to play them well (von Ahn and Dabbish, 2008; Wang et al., 2013), and this generally results in higher quality data (Kuznetsov, 2006). This distinction can be important since oftentimes crowd-sourced datasets are more vulnerable to effects from annotators (Geva et al., 2019). For example, data

quality can suffer if compensation is not appropriate for the task (Fort et al., 2011; Wang et al., 2013), if incentives are misaligned, or in the worst case both (Gneezy and Rustichini, 2000). In our categorization of QA datasets, we include authorship information as well. While there are certainly other factors that influence the distribution of questions in QA datasets, these are some of the most important.

Human-in-the-Loop Adversarial Examples Since the publication of the work in Chapter 6, several new datasets across a variety of NLP tasks have used model-in-the-loop methods to create more challenging datasets. Although it is presumed that datasets like HotPotQA require multi-step reasoning, Min et al. (2019) show that many do not require multi-step reasoning. Dua et al. (2019) and Dasigi et al. (2019) take aim at this problem by not allowing crowd-workers to submit questions answerable by a strong baseline. These works and ours in Chapter 6 create single-round adversarial questions, but the process can be generalized to multiple rounds. After every round of data collection, models are re-trained on the original data plus the adversarial data which makes future questions even more difficult. This procedure has been done for reading comprehension (Bartolo et al., 2020), natural language inference (Nie et al., 2020), and detecting abusive messages in dialog systems (Dinan et al., 2019a). This human-in-the-loop method is one effective way to reduce lexical artifacts in datasets which in turn forces the development of better models (Kaushik et al., 2021). Dynabench (Kielia et al., 2021) draws these and other tasks together into shared evaluation infrastructure that supports iterative adversarial collection. For an extensive survey of these methods, we refer the reader to Wang et al. (2021).

2.3 Question Answering Datasets

A major contributor to the efficacy of recent QA systems is the vast improvement in the availability of large-scale datasets. These datasets and corresponding tasks are extremely diverse, but it is nevertheless helpful to categorize them—where possible—by paradigm (§2.1), answer format (§2.2.1), information context (§2.2.2), generative process (§2.2.3), and domain. With those datasets in mind, Table 2.1 categorizes the most frequently used QA datasets starting from 1999 to the present.²⁵ For further reading, we refer the reader to a QA tutorial by Chen and Yih (2020) or surveys by Cambazoglu et al. (2020) and Rogers et al. (2021).

Dataset	Paradigm	Author	Area	Citation
Deep Read	Manchester		Stories	Hirschman et al. (1999)
TREC-8 QA	Cranfield		News	Voorhees (2000b)
TREC-9 QA	Cranfield		Search	Voorhees (2000a)
TREC QA 2001	Cranfield		Search	Voorhees (2001)
TREC QA 2002	Cranfield		Search	Voorhees (2002a)
TREC QA 2003	Cranfield		Search	Voorhees (2003b)
TREC QA 2004	Cranfield		Search	Voorhees (2004)
TREC QA 2005	Cranfield		Search	Voorhees and Dang (2005)

²⁵In this thesis, we use QA and reading comprehension interchangeably in reference to datasets.

Dataset	Paradigm	Author	Area	Citation
TREC QA 2006	Cranfield	👤	Search	Dang et al. (2006)
TREC QA 2007	Cranfield	👤	Search	Dang et al. (2007)
QA4MRE 2011-2013	Manchester	👤	Multiple	Peñas et al. (2013)
MCTest	Manchester	👤	Stories	Richardson et al. (2013)
WebQuestions	Cranfield	👤+👤	Search	Berant et al. (2013)
CNN/Daily mail	Manchester	👤 ²⁶	News	Hermann et al. (2015)
Simple Questions	Manchester	👤→👤	Freebase	Bordes et al. (2015)
Children’s Book Test	Manchester	👤	Stories	Hill et al. (2016)
bAbi	Manchester	👤	Stories	Weston et al. (2016)
SQUAD 1.0	Manchester	👤	Wiki	Rajpurkar et al. (2016)
WikiReading	Manchester	👤	Wiki	Hewlett et al. (2016)
MS-MARCO	Cranfield	👤	Search	Nguyen et al. (2016)
MovieQA	Manchester	👤	Movies ²⁷	Tapaswi et al. (2016)
RACE	Manchester	👤	Exams	Lai et al. (2017)
TriviaQA	Manchester	👤	Trivia	Joshi et al. (2017)
SearchQA	Manchester	👤	Trivia	Dunn et al. (2017)
Quasar-T	Manchester	👤	Trivia	Dhingra et al. (2017)
SciQ	Manchester	👤↔👤	Science	Welbl et al. (2017)
NewsQA	Cranfield	👤	News	Trischler et al. (2017)
CWQ	Manchester	👤→👤 ²⁸	Wiki	Talmor and Berant (2018)
NarrativeQA	Manchester	👤	Stories	Kočiský et al. (2018)
DuoRC	Manchester	👤	Movies	Saha et al. (2018)
MultiRC	Manchester	👤	Multiple	Khashabi et al. (2018)
HotpotQA	Manchester	👤	Wiki	Yang et al. (2018b)
SQUAD 2.0	Manchester	👤	Wiki	Rajpurkar et al. (2018)
QBLink	Manchester	👤	Trivia	Elgohary et al. (2018)
WikiHop ²⁹	Manchester	👤	Wiki	Welbl et al. (2018)
OpenBookQA	Manchester	👤↔👤	Science	Mihaylov et al. (2018)
QASC	Manchester	👤→👤	Science	Khot et al. (2020)
DROP	Manchester	👤↔👤	Wiki	Dua et al. (2019)
QUOREF	Manchester	👤↔👤	Wiki	Dasigi et al. (2019)
QUAC	Cranfield	👤	Wiki	Choi et al. (2018)
BoolQ	Cranfield	👤	Search	Clark et al. (2019)
ELI-5	Cranfield	👤	Reddit	Fan et al. (2019)
ARC	Manchester	👤	Science	Clark et al. (2018)
Record	Manchester	👤→👤	News	Zhang et al. (2018b)
ROPES	Manchester	👤	Wiki/Sci	Lin et al. (2019)
COQA	Cranfield	👤	Multiple	Reddy et al. (2019)
CosmosQA	Manchester	👤	Narrative	Huang et al. (2019)
Natural Questions	Cranfield	👤	Search/Wiki	Kwiatkowski et al. (2019)
Quizbowl	Manchester	👤	Trivia	Rodriguez et al. (2019)
Trickme	Manchester	👤↔👤	Trivia	Wallace et al. (2019b)
MCScripT	Manchester	👤	Commonsense	Ostermann et al. (2018)
QuaRel	Manchester	👤↔👤	Stories	Taffjord et al. (2019)
CommonSenseQA	Manchester	👤↔👤	Commonsense	Talmor et al. (2019)
AmbigQA	Cranfield	👤	Search	Min et al. (2020)

²⁶ Although noun phrases to mask are created automatically, the source of questions—summary bulletpoints—and the articles are human created.

²⁷ Multi-modal, pairs video and transcripts

²⁸ The original questions are automatically created and then paraphrased by humans.

²⁹ MedHop is a second dataset in this publication.

Dataset	Paradigm	Author	Area	Citation
Curiosity	Cranfield	👤↔️🤖	Geopolitical	Rodriguez et al. (2020)
QuAIL	Manchester	👤	Multiple	Rogers et al. (2020a)

Table 2.1: The table categorizes datasets by the task’s paradigm, the Author (human 👤, machine 🤖, or a mix), and topic domain. To be categorized as human-generated, it is not enough for the dataset to consist of naturally occurring text; it must have questions that are authored by humans instead of being automatically generated. 👤→🤖 and 👤↔️🤖 indicate human-generated questions that are non-trivially checked or modified by a machine system. 🤖→👤 and 🤖↔️👤 indicate machine-generated questions that are paraphrased by humans. 👤↔️🤖 and 👤↔️🤖 indicate question creation incorporates interactive human-machine cooperation.

2.4 Methods for Question Answering

This section reviews methods for QA answering, beginning with those from before the statistical machine learning era and proceeding to modern neural methods.

2.4.1 A Brief History of Symbolic Question Answering Models

Early QA work focused on building systems for narrower, more well-defined, and thus tractable problem domains. Each of these systems developed means of combining knowledge of syntax (grammar), semantics (the meaning of words), and reasoning (ability to deduce and connect facts). BASEBALL showed the ability to manipulate a knowledge-base of baseball games through natural language (Green et al., 1961) which inspired later work in querying databases with natural language (Copestake and Jones, 1990). At a larger scale, LUNAR showed that querying information in natural language—in this case, metadata about Moon mineral samples—was useful to Apollo Program scientists (Woods, 1972). Other work like STUDENT solved simple word algebra problems by parsing numerical expressions and reasoning about relations between them (Bobrow, 1964). QA was also used to show that SHRDLU could manipulate objects in a simulated world (Winograd, 1971). Many other similarly limited symbolic QA systems were developed, but research eventually shifted to the statistical-based methods with the creation of the question answering track at the Text Retrieval Conference (TREC) in 1999. Using the TrecQA track, systems like MULDER (Kwok et al., 2001) showed that QA systems significantly reduced user effort when their information-need is satisfied by short answers. Contemporaneously, systems like PROVERB (Keim et al., 1999) solved crossword problems by combining multiple QA methods with a centralized probabilistic model that decides on the best solution.

2.4.2 Modern Methods for Question Answering

Modern QA systems use either classical methods, deep learning methods, or combine both. The shared goal of both methods is to represent text numerically in a way that can be used to process questions, retrieve relevant documents, and compute an answer. Both families of methods represent text as numerical vectors, but derive their representations differently. Generally, classical methods encode text with a vector space model (Salton et al., 1975) while deep learning methods use distributed word representations (Mikolov et al., 2013). In both cases, the objective is to represent texts such that those that are semantically similar have high similarity scores while those that are semantically different have low similarity scores. This exploits the intuition that the answer to a question—such as “who was the first woman to fly to space”—is more likely be located in a document that is semantically similar to the question than not.

Vector space models compute semantic similarity by considering the overlap between the words mentioned in documents. Intuitively, when terms in a question overlap with terms in a document, it is more likely to contain useful information. The most effective vector space model for QA-motivated IR are Term-Frequency Inverse Document-Frequency TF-IDF (Jones, 1972; Rajaraman and Ullman, 2011) methods like BM25 (Robertson and Walker, 1994). The intuition behind these methods is two-fold: (1) the number of occurrences of a term in a document is correlated with its importance to the document, and (2) the prevalence of a term across all documents is inversely correlated with its importance. In IR and QA, methods like TF-IDF are particularly effective because the words of interest like named entities are precisely those up-weighted.

However, there are two disadvantages to this approach.³⁰ First, although local word order can be encoded by using n-grams instead of unigrams, longer-range word order and thus longer range dependencies cannot be modeled with n-grams.³¹ The second disadvantage is that even with mitigations like stemming, the similarity of related yet unmentioned terms—such as the semantic relevance of “astronaut” to the prior question—is zero. These shortcomings motivated the development of distributed word representations.

Distributed word representations aim to find representations of words such that words mentioned in similar contexts have similar representations. For example, “space” and “astronaut” should have representations that are mathematically similar since they occur in similar contexts. Concretely, for a corpus comprised of V unique words, each is represented as a row in a $d \times V$ -dimensional matrix \mathbf{W}_t . Conceptually, one might imagine that each of the d dimensions encodes semantic aspects of each word like tense, plurality, or “spaciness” although none of these are defined or guaranteed a priori.

Terms are considered more similar if their dot product is more positive and less

³⁰Additionally, out-of-vocabulary issues are also amplified in QA due to the prevalence of named entities.

³¹The number of unique n-grams $O(|V|^n)$ is exponential with respect to a vocabulary V , thus the probability in natural text of any given n-gram exponentially decays with respect to n .

similar otherwise. Since the numerical representations of words are not determined a priori, they must be trained. For example, Word2Vec (Mikolov et al., 2013) or GLoVE (Pennington et al., 2014) are algorithms that create representations such that words that appear in similar contexts have similar representations. Word similarity can be extended to sentence similarity by aggregating word representations through operations like summation, averaging, or maxout. There still remains at least two drawbacks to distributed word representations: (1) during inference, word order and more generally context *still* is not factored in (e.g., “apple” the fruit versus the company is context-dependent), and (2) after training, these representations are not further updated (fine-tuned) based on the target downstream task.

Order-aware neural networks address both these issues. These models begin by representing the tokens $\mathbf{w} = [w^{(1)}, \dots, w^{(n)}]$ with vector embeddings $[\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(n)}]$ and then build context-aware representations $[\mathbf{h}^{(1)}, \dots, \mathbf{h}^{(n)}]$ that the downstream task uses. Section 2.4.3 describes several order-aware models for building these representations. Pre-trained word embeddings are also a powerful mechanism for transfer learning. Similarly, initializing order-aware models with parameters from pre-trained neural language models (McCann et al., 2017; Peters et al., 2017, 2018; Devlin et al., 2018) is an effective way to improve downstream tasks. Lastly, since these models are fully differentiable, as long as the end task (or a suitable proxy) is expressible by a loss function, the full model can be fine-tuned to the end task. Although these models have some downsides like difficulty to train and computational cost, the improvements in metrics like accuracy and recall are well worth the tradeoff.

Modern QA systems combine some or all of these textual representations to find documents relevant to the question and to answer it. The most common, general architecture for QA breaks the problem into two components: (1) an optional document retrieval system to find passages relevant to the question, and (2) a system that—possibly using the retrieved documents—derives a final answer. Answers can take multiple forms like free-response, choosing from a closed set, multiple-choice, and span selection. These answers are used along with an evaluation measure to give the system feedback during training.

This architecture is advantageous for three reasons. First, it logically separates two related but distinct problems: collecting documents likely to contain relevant information to the question and reasoning about the contents to arrive at an answer. This architectural inductive bias mimics one human approach to information-seeking: we rapidly collect seemingly relevant documents (e.g., via Google search or by asking a librarian) then read those documents more carefully to find the answer. Thus, these sub-problems can be studied in isolation first and then composed and improved together later. Second, this approach makes the QA tractable even when there are millions or billions of documents. For both humans and machines, reading a document with a specific question in mind is computationally expensive; doing this process for every possible document is intractable and computationally wasteful since only a small fraction of documents are relevant for any given question. By separating these problems, the document retrieval step can be designed to be scalable and efficient through classical approaches like inverted indices (Zobel and Moffat,

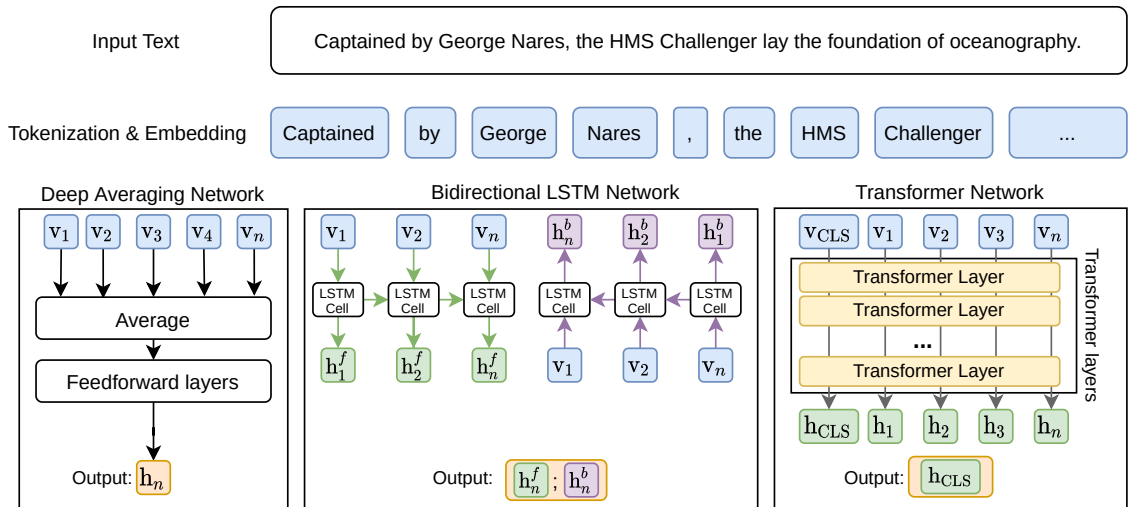


Figure 2.8: Deep averaging networks, recurrent neural networks (e.g., LSTMs and GRUs), and transformer networks (e.g., BERT) are common architectures in neural NLP models. To produce text representations of a fixed size (i.e., not a sequence of vectors), each architecture takes a different approach. Deep averaging networks average word vectors and then pass the subsequent vector through one or more feedforward layers; recurrent models often concatenate the last hidden state in each direction; transformer networks usually have a special token associated with the input’s general representation (in BERT, the CLS token).

2006) or efficient nearest neighbor search over numerical vector representations (Lee et al., 2019). Third, this format fits a variety of tasks without additional work; QB uses only step two,³² SQuAD uses both steps, but the document is provided, and Open Domain QA (Chen et al., 2017) uses both. Combined, these benefits have made this two-step approach common for modern QA (Chen and Yih, 2020).

Next we describe specific neural methods for neural text representation (Section 2.4.3), classical and differentiable document retrievers (Section 2.4.4), and neural answering models (Section 2.4.6).

2.4.3 Neural Text Encoders

This section reviews several common neural text encoding methods used in IR and QA models. These encoders create question and document representations that models then combine to derive an answer. At a high level, these methods aim to represent text (questions, documents, or answers) so that semantically similar concepts have similar encodings.

Given text t with tokens $[w^{(1)}, \dots, w^{(n)}]$, a text encoder should return a sequence of representations $[\mathbf{h}^{(1)}, \dots, \mathbf{h}^{(n)}]$ —one for each token.³³ For tasks where

³²Finding relevant documents could aid prediction, but we do not provide human-approved annotations.

³³We use the term token embedding since it can combine the word and character embeddings

predictions are not per-token (e.g., predicting the start of a span), encoders should also define an aggregated representation $\mathbf{h}^{(*)}$. Despite compressing the meaning of a text to a fixed-length vector, the aggregated representation still captures substantial semantic information (Conneau et al., 2018). We assume that the per-token representations, aggregate representation, or both are ultimately incorporated into the model’s loss function which is then optimized via backpropagation and stochastic gradient descent (Rumelhart et al., 1986; Goodfellow et al., 2016). Figure 2.8 shows several common text encoders in NLP: deep averaging networks, recurrent architectures, and transformer architectures (§5.5.3).³⁴

Word embeddings are the simplest approach where each word in a vocabulary of size V is represented by a d -dimensional vector \mathbf{v}_i of trainable parameters. In addition to representing distinct terms, token embeddings can be replaced by character embeddings (Kim et al., 2016a) or used in addition to them (Joulin et al., 2017). While the parameters can be trained from scratch on the target task training data, it is also common to pre-initialize them with weights from Word2Vec or GLoVe and then update these representations; this is particularly effective when there is scarce training data for the target task.

Like “frozen” word embeddings, word sequences can be aggregated to $\mathbf{h}^{(*)}$ through operations like average pooling

$$\mathbf{h}^{(*)} = \frac{1}{n} \sum_{i=1}^n \mathbf{v}^{(i)} \quad (2.1)$$

and max pooling

$$\mathbf{h}^{(*)} = \frac{1}{n} \max_{i=1}^n \mathbf{v}^{(i)} \quad (2.2)$$

the word vectors. Alternatively, the word vectors $\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(n)}$ can be passed through one or more feedforward layers

$$\mathbf{h}_0 = \frac{1}{n} \sum_{i=1}^n \mathbf{v}^{(i)} \quad (2.3)$$

$$\mathbf{h}_n = f(\mathbf{W}_{n-1} \cdot \mathbf{h}_{n-1} + \mathbf{b}_{n-1}) \quad (2.4)$$

with parameters $\mathbf{W}_n, \mathbf{b}_n$ and a non-linearity f as in a deep averaging network (Iyyer et al., 2015, DAN). Although word embeddings alone and DANs are unordered syntactic representations, they are still effective for some QA and sentiment analysis tasks where word order is less important and attractive in terms of speed and computational budget.

Next, we consider other composition functions. Throughout this thesis, we adopt the notation that given a sequence of tokens $\mathbf{w} = [w_1, \dots, w_n]$ and a composition function ENCODER,

$$\text{ENCODER}^{(i)}(\mathbf{w}) \quad (2.5)$$

for a specific token.

³⁴We do not cover convolutional architectures since they are less common for NLP.

denotes the encoder’s output representation of word $w^{(i)}$ and

$$\text{ENCODER}(\mathbf{w}) \tag{2.6}$$

denotes the aggregated representation—as defined by the model. For example, $\text{DAN}(\mathbf{w}) = \mathbf{h}_n$ is the aggregated DAN representation and $\text{LSTM}^{(i)}(\mathbf{w})$ is an LSTM’s contextualized representation of the i th input token (§5.5.3). In BERT-based models, we amend this slightly and will refer to the representation of the CLS token as $\text{BERT}^{\text{CLS}}(\mathbf{w})$.

The most common order-aware neural composition functions are simple recurrent networks (Rumelhart et al., 1986), long short-term memory networks (Hochreiter and Schmidhuber, 1997, LSTM), gated recurrent networks (Cho et al., 2014b, GRU), and transformer networks (Vaswani et al., 2017). Recurrent representations (RNNs, LSTMs, and GRUs) represent sequences (in the first layer) sequentially: each word’s representation is conditioned on all previous words. For tasks where representation in both directions is important—such as text span selection—the opposite direction can be modeled by concatenating the original representation with the representation of the reversed sequence (bidirectional) (Schuster and Paliwal, 1997). Additionally, these layers can be stacked on top of each other; empirically, layers deeper in the network have more abstract representations (Peters et al., 2018). Transformer networks—especially large-pretrained language models like BERT (Devlin et al., 2018)—are another powerful composition function. Unlike recurrent networks, these transformers are comprised of alternating attention and feedforward layers, and as a consequence the representation of every word is (symmetrically) conditioned on the representation of every word in the prior layer. While these models can be challenging to train due to training dynamics, computation speed, and memory consumption, they are the foundations of most state-of-the-art models in QA. For a survey of BERT-related work see Rogers et al. (2020b). Next, we describe how these text encoders are used to construct document retrievers and question answering modules.

2.4.4 Document Retrieval

The first component of a two-stage QA system is the document retriever. The responsibility of the document retrieval module is to return—from a corpus of documents $\mathcal{D} = \{d_1, \dots, d_n\}$ —a small list of n_k documents \mathcal{D}_k or even only one that is highly relevant to a question q .

In QA systems, document retrievers play the role of librarians. It would be infeasible for a librarian to carefully read every book in a library every time a patron asked a question; instead, librarians categorize and index libraries so that they can quickly provide general suggestions. The objective of these systems is to design a document encoding $E_d(d_i)$, a question encoding $E_q(q)$, and scoring function f such that a scalar score

$$s_i = f(E_q(q), E_d(d_i)) \tag{2.7}$$

is correlated with labeled relevance judgments. For either a human or machine librarian, the primary constraint in retrieving documents is the computational cost

incurred for resolving a query q as the number of documents n_d in the corpus grows. In practice, document collections can include hundreds of thousands of news articles (Voorhees, 2000b), millions of Wikipedia pages, or billions of webpages (Craw). Thus, commonly used document retrievers first *pre-index* \mathcal{D} and then at inference time use that index to efficiently return a list of the n_k documents with highest scores s_i . Next, we describe the classical and neural approaches used to define the encoders and scoring functions.

The predominant method for classical document retrieval indexes documents as bag-of-word (Jones, 1972) using a TF-IDF (Rajaraman and Ullman, 2011) encoder E_{tf-idf} , stores them in an inverted index (Zobel and Moffat, 2006), and uses a BM25 scoring function f_{bm25} . For QA, the terms valued by TF-IDF are usually correlated with judgements. In Chapters 5 and 6, we use BM25

$$\sum_i IDF(q^{(i)}) \frac{g(q^{(i)}, D) \cdot (k_1 + 1)}{g(q^{(i)}, D) + k_1 \cdot (1 - b + b \cdot \frac{fieldLen}{avgFieldLen})} \quad (2.8)$$

$$IDF(q^{(i)}) = \ln\left(1 + \frac{docCount - f(q^{(i)}) + .5}{g(q^{(i)}) + .5}\right) \quad (2.9)$$

scoring for document retrieval. The function $g(q_i, D)$ represents how many times the token $q^{(i)}$ occurs in document D and $g(q^{(i)})$ is the number of documents that contain token $q^{(i)}$.³⁵ Apache Lucene (Foundation) and Elasticsearch (Gormley and Tong, 2015) are two industry-standard, user-friendly, and scalable implementations of this document retriever. DrQA (Chen et al., 2017) show this approach is effective for Open-domain QA and we show in Chapter 5 that it is a strong baseline for QB.

2.4.5 Neural Document Retrieval

Neural networks have also proven to be effective document retrievers, and their design has evolved from augmenting the output of a classical system to replacing it entirely (Mittra and Craswell, 2018).

Initial neural systems do not replace the retrieved document set \mathcal{D}_k , but instead re-rank its results (Wang et al., 2018). If the underlying system has high recall at n_k , but low precision in the first few documents then this method is effective. Neural networks can also adaptively determine an appropriate number of documents to retrieve to reduce noise in the results (Kratzwald and Feuerriegel, 2018). Other work uses neural networks to reformulate queries to a classical IR retriever (Buck et al., 2018) or even iteratively retrieve documents (Das et al., 2018). As effective as they are, these systems are restricted by the accuracy and recall of the underlying, black box IR system.

Current state-of-the-art in document retrieval replaces classical systems with an end-to-end neural approach. Like classical systems, neural retrievers define a similarity score $s_i = f_n(E_d(d_i), E_q(q))$ that factorizes the representation of the documents and questions. This factorization allows all the document encodings to be

³⁵ b and k_1 are hyper parameters.

pre-computed once training is complete. The primary challenges are defining the scoring function in a way that is computationally scalable in n_d , defining the form of the encoders, and defining the loss function that encoder parameters are trained with.

Most document retrievers encode questions and documents with a text encoders from Section 2.4.3. For example, ORQA (Lee et al., 2019), REALM (Guu et al., 2020), DPR (Karpukhin et al., 2020), RAG (Lewis et al., 2020b), and ColBERT (Khattab and Zaharia, 2020) use the BERT architecture for both E_q and E_d and later fine-tune it with a retrieval-specific loss function. Similarly, all of these models use maximum inner product search (Ram and Gray, 2012, MIPS)—typically just the dot product—as the similarity function. Using DPR as an example, the question encoding BERT(“what was the first Mars rover?”) is compared using the dot product to every Wikipedia passage encoding BERT(d)—including one mentioning the answer “Sojourner”; The highest scoring passage is returned. Exhaustive calculation of dot product similarity is efficient even across all documents D (Abuzaid et al., 2019) and can be made more efficient through approximate nearest neighbor search (Johnson et al., 2019). Next, we describe how the retriever’s parameters are trained.

Although models vary in exact training procedure, the general approach is to define a metric learning loss based on the score and relevance label of the document (Kulis, 2012). For example, the loss may be binary cross-entropy using a binary relevance label and a (probability) score ranging from zero to one. In particular, this framing imposes the inductive bias that the document and question encodings should have a high score $f_n(E_d(d_i), E_q(q))$ when the document is labeled as relevant. The labels for document relevance may be defined in the task itself (e.g., SQuAD or NQ) or automatically generated like ORQA’s Inverse Cloze Task (Lee et al., 2019). While this general approach is not the only way to train these encoders, it is one effective and general-purpose approach. The next component of the two-stage QA system is the model for deriving answers from the question and/or document.

2.4.6 Neural Answering Module

The role of the answering module is to—given the question and (optionally) relevant documents—return the most likely answer. The form of the answers depends on the task; for QB it is a Wikipedia entity, for SQuAD and NQ a text span, and for open QA free text. Although the documents for the answer module may already have an encoding from the classical or neural retriever, most answer modules re-encode the question, document, or both. This is desirable since the retriever representations of the question and document are independent of one another and therefore not particularly expressive. Here we describe neural architectures and loss functions for (1) models that take only the question as input and (2) that take the question and a document as input.

The simplest architecture for QB is classification over the answer classes (distinct Wikipedia titles). As in the retriever, the question can be represented by any of the text encoders in Section 2.4.3. Following this, the representation can be passed to any number of hidden layers before being projected to the classification

dimension. This output layer has logits corresponding to each answer label, and the model is trained with cross-entropy loss. For further details on this type of model, we refer the reader to Section 5.5.

For tasks like SQuAD and NQ (or QB with supporting evidence like TriviaQA), the prediction is the shortest text span in the document most likely to contain the answer. Thus, one goal of re-encodings in these tasks is to jointly encode the question and document to identify tokens that are most relevant to the answer. For example, BIDAf (Seo et al., 2017) adds question-to-document and document-to-question attention mechanisms; interpreting the document with the question “in mind” has shown to be important in QA (Weissenborn et al., 2017). More recent architectures like transformers jointly model questions and documents through all layers. Other models may jointly model this and additional information such as graph connections (Zhao et al., 2020a,b). In all of these models, the sequence of outputs is used to classify whether each token is the start or end of the correct answer span.

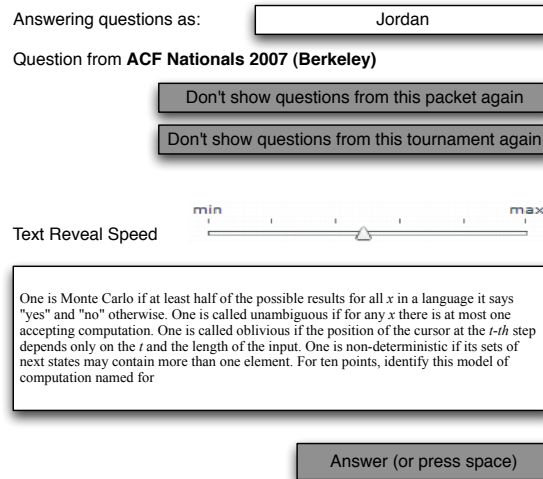
Throughout this thesis we use classical and neural IR and QA as parts of dialog systems (Chapter 3), systems that play QB (Chapter 5), and a human-in-the-loop question writing interface (Chapter 6). Before moving to this work, we briefly review prior work in QB that laid the foundation for Chapters 5 and 6.

2.5 Prior and Related Work in Quizbowl

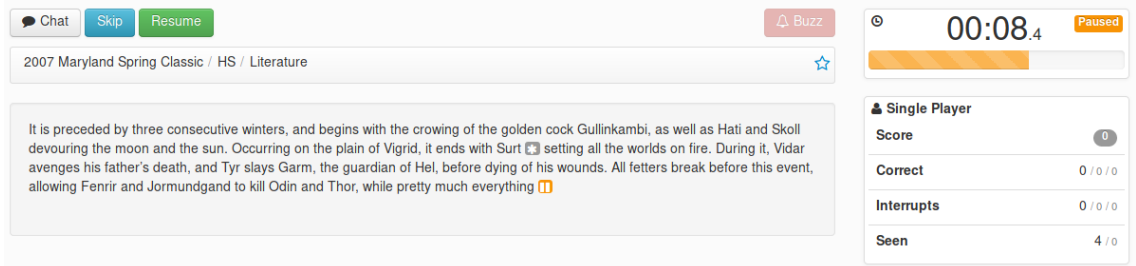
Although QB has been played for decades, the first online interface was only created in 2012 (Figure 2.9a). After a day online, over 7000 games were played, and by the end of the two-week data collection period, 43000 questions were played by 461 users (Boyd-Graber et al., 2012). After the initial interface was taken offline, enterprising members of the QB community reacted to the sudden deprivation by resurrecting the interface. Over the years, its successor Protobowl (<http://protobowl.com>) collected orders of magnitude more data and built a competitive experience by adding multiplayer support (He et al., 2016b).

On both platforms, users play questions from prior tournaments: words in the question are revealed one-by-one until the player attempts to answer the question. Every time a question is played, the word that the player buzzed on, their answer, and whether their answer was correct are recorded. In Chapter 5, we discuss how updated this dataset to include 3.9 million records from over ten thousand users.

The QB-related work in this thesis uses “tossup” questions, but there are other types such as bonus questions as in QbLink. Each bonus question contains multiple parts, and the player answers each one-by-one. Bonus questions reward multi-step reasoning by making later questions dependent on correctly answering previous questions. QbLink (Elgohary et al., 2018) builds a dataset of these bonus questions. While QB and its educational focus are a distinctly positive example of intellectual competition, the measurement of human intelligence more generally has not always a net benefit to society.



(a) The 2012 interface was the first way to play QB online (Boyd-Graber et al., 2012).



(b) The QB interface from He et al. (2016b) for collecting gameplay. Beyond using modern web frameworks, it also allows for real-time competitive play and chat with other players.

Figure 2.9: The original QB interface and a popular modern interface for playing QB online. Both interfaces reveal questions word-by-word until a player interrupts the system, and makes a guess.

2.6 The Harrowing Past of Measuring Human Intelligence

This thesis takes inspiration for measuring machine intelligence from how human intelligence has been measured in psychometrics,³⁶ a field concerned with the measurement of intelligence, skill, and abilities in humans (Guilford, 1954). Therefore, it feels necessary to acknowledge and raise awareness of how measurement of human intelligence was used to justify eugenic and racist policies that contributed to horrific human rights violations and discuss how this history is relevant to the present day. At least two pioneers of psychometrics, Francis Galton and J. McKeen Cattell (Jones and Thissen, 2006), were eugenicists that held repugnant views advocating that society “welcome and support the eugenic movement tending to limit the birth of feeble-minded and defective children” (Cattell, 1915). In the United States, the legal implementation of forced sterilization of the “feeble minded” was endorsed by the U.S. Supreme Court in *Buck v. Bell* (Court, 1927; Gross, 2016). Similar justifications were employed by 30 states in legislation that led to the sterilization 60,000 U.S. citizens across 30 states (Norrgard, 2008) including Puerto Rico where “about one third of women of childbearing age, a figure that remained constant through the 1980s,” were sterilized (Briggs, 2003). Eventually, “America’s eugenic movement spread to Germany as well, where it caught the fascination of Adolf Hitler and the Nazi movement,” directly contributing to the Holocaust (Black, 2012).

Faced with the need and opportunity to “scientifically” support unscientific, eugenic views like scientific racism (Gould, 1981), “eugenicists labored to devise objective methods of measuring and quantifying valued traits, including intelligence” (Reddy, 2007). The Alpha and Beta group intelligence tests of WW1 are the first example of large-scale intelligence tests that directly influenced access to opportunities (for military recruits in this case). These tests “became models for future group tests” (Spring, 1972) despite the fact that a developer of these tests, the eugenicist Henry H. Goddard, later admitted “we do not know what intelligence is” (Goddard, 1920). The same intelligence tests were later used to support the eugenic Immigration Act of 1924 (Gelb et al., 1986; Jacobson, 1999, p. 83–85).³⁷ While the Alpha and Beta intelligence tests—created and promoted by eugenicist Carl C. Brigham—are lesser known to the common person, their immediate intellectual descendant is known to nearly every college student as the Scholastic Aptitude Test or SAT (Lemann, 2000; Black, 2012).³⁸ Despite this inter-connected history, some still (falsely) claim that “intelligence tests are not culturally biased against American blacks or other native-born, English-speaking peoples” (Arvey et al., 1994; Goldstein, 2012). As historian and scholar Ibram X. Kendi writes,

Standardized tests became the newest “objective” method of proving Black intellectual inferiority and justifying discrimination, and a multimillion-

³⁶For example, Item Response Theory (4) was originally developed in psychometrics and is used in Chapter 4.

³⁷The Immigration Act severely restricted immigration to the United States “to preserve the ideal of U.S. homogeneity” (U.S. Department of State Office of the Historian, 2021).

³⁸While Brigham later recanted his view that the SAT could measure intelligence, substantial irreparable damage had already been done by fueling eugenic discourse.

dollar testing industry quickly developed in schools and workplaces (Kendi, 2016, p. 310).

Since then, Kendi has called standardized tests “the most effective racist weapons ever devised to objectively degrade Black and Brown minds” (Kendi, 2020) and some in psychology have argued that “the validity of IQ tests is questionable” (Weiten, 2016, p. 281).

In a world where numbers—such as the SAT (and other intelligence tests)—and algorithms play crucial roles in structuring society (O’Neil, 2016, p. 134), it is crucial to consider the *role* that technology plays in society. The evolution of psychometrics and the standardized testing industry is one example of a “New Jim Code: the employment of new technologies that reflect and reproduce existing inequities but that are promoted and perceived as more objective or progressive than the discriminatory system of a previous era” (Benjamin, 2019, p. 5).³⁹ This and numerous other examples emphasize that,

No object or algorithm is ever either good or evil itself. It’s how they’re used that matters...Forming an opinion on an algorithm means understanding the relationship between human and machine...It’s about asking if an algorithm is having a net benefit on society (Fry, 2018, p. 3).

In the context of testing machine intelligence, the goals of this thesis are to use this technology to better serve humans seeking to expand their knowledge (e.g., Chapter 6) and improve how we compare the effectiveness of machines on QA tasks (e.g., Chapters 4). As we highlight in Chapter 7, we should—as a field—critically think about the ways that incentive structures in benchmarks influence the type of algorithms and technology rewarded with recognition and use these structures as a way to reward algorithms that are a net benefit to society. Finally, as we discuss multiple times in this thesis, comparisons between humans and machines on QA tasks should *not* be used to make claims of the intellectual superiority of one over the other.

In the next chapter, we revisit the Cranfield paradigm and introduce a new dataset for conversational information-seeking. Following this, Chapter 5 describes the traditions of QB as a trivia game, frames it as a machine learning task, builds models to play the game, and evaluates them with offline metrics and live exhibition matches. Afterward, Chapter 6 introduces a methodology for centaur authoring of adversarial questions. We conclude by outlining directions for future work.

³⁹Benjamin’s coining of the term a “New Jim Code draws on *The New Jim Crow*, Michelle Alexander’s (2012) book that makes the case for how the US carceral system...permits legalized discrimination” (Benjamin, 2019, p. 8).

Chapter 3: Information Seeking in the Spirit of Learning: A Dataset for Conversational Curiosity

So great is our innate love of learning and of knowledge that no one can doubt that man’s nature is strongly attracted to these things even without the lure of any profit.

Marcus Tullius [Cicero](#), 45BCE

Conversations are a natural form for humans to seek information, and there are decades of study on formal dialogues and interactions of users with reference librarians. The natural next step is to design automated systems that are ‘virtual librarians’, eliciting information needs, correcting misconceptions, and providing the right amount of information at the right time across all possible domains.

Research Frontiers in Information Retrieval:
Report from the Third Strategic Workshop on
Information Retrieval ([Culpepper et al., 2018](#))

With working definitions of the Cranfield and Winograd paradigms in hand, we introduce a dataset for conversational information-seeking that whose evaluation follows the Cranfield paradigm. This work contributes a large resource of 14K dialogs to the nascent area of conversational information-seeking ([Dalton et al., 2020](#)), investigates various means to evaluate the effectiveness of dialog strategies, and proposes a baseline model for the dataset’s dialog task.¹ At the heart of the task is a dialog between a user and an assistant—role-played by human annotators—where the user is encouraged to engage in curiosity-driven inquiry about a geopolitical entity. We design dialog interfaces so that while they converse, we record information that characterizes the assistant’s dialog strategies and measure the user’s engagement both directly and indirectly. With this data, we validate that answers to user questions that build on pre-existing knowledge tend to increase engagement as measured by the number of times that the user asks a followup question. In the next chapter, we contrast this dataset’s goal of increasing user engagement with Quizbowl’s goal of testing knowledge.

¹This chapter is based on the EMNLP publication [Rodriguez et al. \(2020\)](#).

U: <assistant wake-word>, tell me about Tahiti.
A: It's the largest island in French Polynesia, near the center of the Pacific
U: What is its history with France?

Figure 3.1: An example of information-seeking dialog that the Curiosity dataset aims to support. Assistants should answer user questions and convey information that inspires meaningful followup questions.

3.1 Motivation for Conversational Information-Seeking

Humans are naturally epistemically curious (Berlyne, 1954) and conversational agents—such as Alexa, Siri, and Google Assistant—should encourage this by helping users discover, learn, and retain novel factual information. More generally, systems for conversational information-seeking should help users develop their information need, be mixed-initiative, incorporate user memory, and reason about the utility of retrieved information as a combined set (Radlinski and Craswell, 2017). We focus on a curiosity-driven, information-seeking scenario where a user starts a conversation with an assistant by asking an open-ended question and then drills down into interest areas (Figure 3.1).

In this setting, what policies should assistants pursue to maintain the user’s interest in the topic? Theories of human learning, such as Vygotsky’s zone of proximal development, posit that learning novel information should be rooted in pre-existing knowledge and skills of the learner (Chaiklin, 2003). Considering this, a good policy may give general information about Tahiti; a better policy would select information related to the user’s knowledge (e.g., familiarity with France); along similar lines, Jennings (2006, p. 81) articulates that “if you already know a few little bits... the new information has something to cling to, barnacle-like, in your synapses.” We hypothesize that engagement and by proxy satisfaction is correlated with policies that integrate a user’s pre-existing knowledge, and we test this through a large-scale, Wizard-of-Oz (WoZ) style collection (Kelley, 1984; Wen et al., 2017) that captures assistant policies, user reactions, and topically relevant entities that the user knows about. This dataset—the Curiosity dataset—has 14,048 English dialogs annotated with sentence-level knowledge grounding, the user’s prior knowledge, dialog acts per message, and binary ratings per message.²

In our dialog task (Figure 3.2), one worker takes the role of a curious user learning about a geographic entity and the other of a digital assistant with access to Wikipedia facts (§3.2). At the start of each dialog, the user is assigned an entity as their topic (e.g., Puerto Rico) along with two *aspects* (e.g., *history* and *demographics*) to investigate. Beforehand, we show the user a list of entities related to the topic, and they mark which they know; these entities are a sample of their pre-existing knowledge. The user engages in open-ended discovery while the assistant simultaneously answers the user’s questions and proactively introducing facts likely

²Dataset and code at curiosity.pedro.ai.

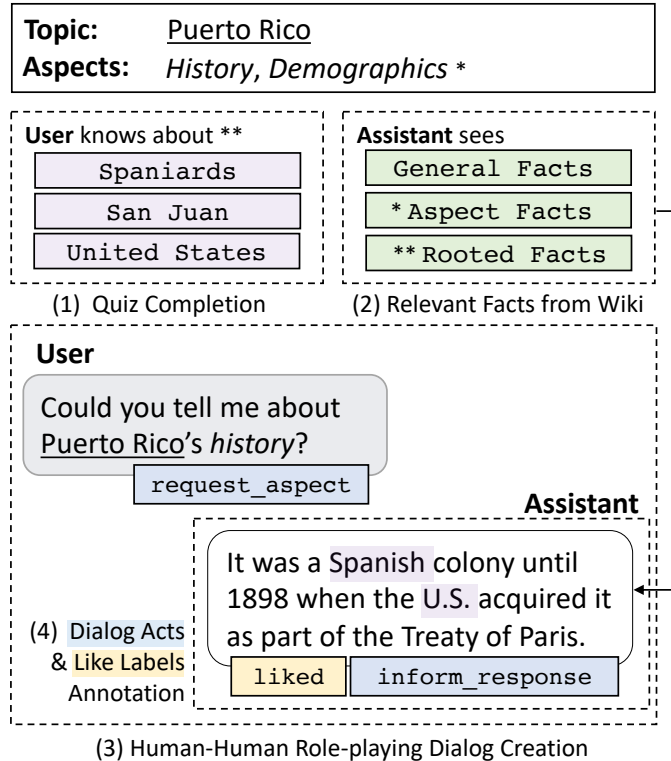


Figure 3.2: We sample pre-existing knowledge by asking users to indicate which topically related entities they already know. The assistant paraphrases facts related to either known entities (rooted facts), an aspect (aspect facts), or the topic generally (general facts). The user expresses engagement through a like button. Dialog acts are annotated in a separate crowd-source task.

to prompt followup questions. For example, if the assistant knew of a user’s familiarity with astronomy when providing information about Puerto Rico, then the user is more likely to engage with and remember facts about the Arecibo Observatory.

Although our dialog task methodology differs from typical Cranfield evaluations (§2.1.1), we share the same end goal to satisfy users. Section 3.3 uses dialog act annotations combined with explicit and implicit user feedback to compare assistants’ content selection and presentation policies. For example, in interactions where the user asks a question and the assistant paraphrases a fact, how often does the user ask a followup question versus trail off in disinterest? Most datasets (Section 3.6) do not have enough annotations to answer these questions since it requires message-level dialog act annotations and feedback signals. We compare three assistant policies: using a fact with a rooted entity, a fact from the user’s aspect, and a generic fact about the topic. The policies are compared through user “likes” of assistant messages and by the dialog act of their subsequent message (e.g., did they ask a specific followup or change topic).

While the focus of this work is in creating the Curiosity dataset and a methodology for measuring user interest, we also introduce models that attempt to mimic assistant policies. These models predict what type of message to send and which

fact to use (§3.4), but do not generate dialog text. Since there are multiple facets to the assistant’s actions, we jointly train models with a multi-task objective function. We compare an end-to-end BERT (Devlin et al., 2018) model to our task-specific Hierarchical Recurrent Encoder model (Serban et al., 2015) and show that our model (CHARM) improves over the baseline.

In summary, this chapter makes three main contributions: (1) we design an experiment to test the efficacy of personalizing conversational information systems through a user’s prior knowledge, (2) introduce the Curiosity dataset—the first dialog dataset combining sentence-level knowledge groundings, per message ratings, and per message dialog act annotations, allowing for robust and fine-grained structural learning of dialog policies for similar applications, and (3) design a multi-task model that incorporates the user’s prior knowledge and improves over a BERT baseline.

3.2 Building the Curiosity Dataset

This section describes the construction of the Curiosity dataset. Dialog topics consist of prominent world geographic entities. The *worldwide* spread of entities makes each novel to most users, the consistent topic type makes starting dialogs easier, and their rich histories, demographics, and economics add topical diversity. For example, most people are only vaguely familiar with the history of Puerto Rico, but most know about related concepts such as the United States or Hurricane Maria. Section 3.2.1 describes how we select geographic topics, aspects, and derive a set of facts to ground against. We collected the dataset in two steps: (1) collecting dialogs with a custom interface (Section 3.2.2) and (2) after-the-fact dialog act annotation (Section 3.2.3). Sample dialogs from the Curiosity dataset are in Appendix A.3.

3.2.1 Choosing the Geographic Topics, Aspects, and Facts for the Dataset

We select 361 geographic pages from Wikipedia that have separate geography and history pages (e.g., Puerto Rico, Geography of Puerto Rico, and History of Puerto Rico).³ We use sentences from each page to build a set of 93,845 facts. We run an entity linker over the content (Gupta et al., 2017) and index each fact by its source page (*topic*), source section (*aspect*), and mentioned entities. Finally, we fit a TF-IDF text matcher (Rajaraman and Ullman, 2011) with Scikit-Learn (Pedregosa et al., 2011). While conversing, assistants are shown facts filtered by topic, aspect, or mentioned entities, that are ranked by textual similarity to the dialog.

3.2.2 User and Assistant Dialog Interfaces

To collect dialogs, we build user and assistant interfaces for annotators. The user’s interface samples their prior knowledge of a topic, captures which assistant

³The existence of a page implies that the topic has ample historical and geographical knowledge to draw from.

Your goal is to learn about Lesotho
Epecially about its "Culture" and "History".

Completing this Quiz is VERY important!
 It helps the assistant answer your questions

Check boxes if

1. **Geography:** if you could **locate** it on a map
2. **Concept:** if you could accurately **explain** what it is

When done, tell the assistant what you want to learn about

Related Entities

Entity	Do you know
Pretoria	<input type="checkbox"/>
Sotho people	<input type="checkbox"/>
United States	<input type="checkbox"/>
Temple Mount	<input type="checkbox"/>
Mohale's Hoek District	<input type="checkbox"/>
South Africa	<input type="checkbox"/>
Orange Free State	<input type="checkbox"/>
Basutoland	<input type="checkbox"/>
Book of Common Prayer	<input type="checkbox"/>
Africa	<input type="checkbox"/>
United Kingdom	<input type="checkbox"/>
Asia-Pacific Economic Cooperation	<input type="checkbox"/>

Figure 3.3: In this example, the user is assigned to learn about Lesotho, specifically its *culture* and *history*. Given their topic, we users indicate which (related) entities—related to Lesotho—are familiar. Related entities range from relatively common like the United States to lesser known like Basutoland. We also provide guidelines and videos before crowd-workers start working on the task.

messages interest them, and manages the dialog context. The assistant’s interface provides contextually relevant facts. Appendix A.1 has screenshots and details of each interface component.

Sampling User’s Prior Knowledge When deployed, digital assistants can draw from prior interactions (Raman et al., 2014) to estimate what a user knows. However, since we do not have these prior interactions, we collect information about what users know. Instead of exhaustively asking about every entity related to the topic, we sample this knowledge. Before the dialog begins, we show the user fifteen related entities that range from commonplace to obscure (United States versus Taíno). Users mark the entities they could (1) locate on a map or (2) summarize succinctly in one sentence (Figure 3.3).



Figure 3.4: The user expresses the “interestingness” of the assistant’s messages through a “like” button (right of message). This is one of the two ways that we estimate user satisfaction.

Like Button for User Interest As part of our collection, we aimed to determine what fact-grounded utterances users found interesting. Users were told to “like” the assistant’s message if they found it “interesting, informative, and relevant to their topic” (Figure 3.4). These likes are an explicit means to estimate user satisfaction.

Assistant’s Topic Summary and Fact Bank The worldwide spread of Curiosity’s entities makes them unfamiliar to most crowd-workers, including the assistants. So that the assistant can still engage the user, the assistant interface provides contextually relevant information. First, the interface shows a topic summary from Wikipedia. Second, the assistant paraphrases facts from a contextually updated fact bank (box 2 in Figure 3.2 and Figure 3.5). To reduce information overload, we use simplified topic descriptions from SimpleWikipedia and show a maximum of nine facts at a time.⁴ We encourage assistants to “stimulate user interest and relate information to things they already know or have expressed interest in.” Assistants are instructed to select relevant facts, click the “use” button, and paraphrase the content into their next utterance.

Like [Dinan et al. \(2019b\)](#), the fact bank selects facts to the assistant using TF-IDF textual similarity (§2.4.4) to recent dialog turns but differs by incorporating the user’s prior knowledge. We show the assistant nine facts: three facts that mention an entity familiar to the user (rooted facts), three facts from their assigned aspects (aspect facts), and three from anywhere on the page (general facts). By construction, rooted facts overlap with the exclusive categories of aspect and general facts. For each category, we find the nine highest TF-IDF scoring facts and then randomize their order. To avoid biasing the assistant, we do not inform them about the user’s known entities or distinguish between types of facts.

3.2.3 Dialog Act Annotation

Inducing structure on conversations through dialog acts is helpful for analysis and downstream models ([Tanaka et al., 2019](#)). We introduce structure—beyond

⁴If a description exists in simple.wikipedia.org, we use that; otherwise, we use the description from en.wikipedia.org.

Facts			
Entity	Section	Fact	
Anchorage, Alaska	Economy	Military bases are a significant component of the economy in the Fairbanks North Star , Anchorage and Kodiak Island boroughs , as well as Kodiak .	Fact Used
United States	Economy	The Trans-Alaska Pipeline can transport and pump up to 2.1 Moilbbl of crude oil per day , more than any other crude oil pipeline in the United States .	Fact Used
United States	Body	United States armed forces bases and tourism are also a significant part of the economy .	Fact Used
Alaska Natives	Economy	Many Alaskans take advantage of salmon seasons to harvest portions of their household diet while fishing for subsistence , as well as sport .	Click to Use Fact
Yakutat, Alaska	Cities, towns and boroughs	Yakutat City , Sitka , Juneau , and Anchorage are the four largest cities in the U.S. by area .	Click to Use Fact
Unorganized Borough, Alaska	Cities, towns and boroughs	The remaining population was scattered throughout Alaska , both within organized boroughs and in the Unorganized Borough , in largely remote areas .	Click to Use Fact
Oregon Territorial Legislature	State symbols	The willow ptarmigan is common in much of Alaska . , State fish : king salmon , adopted 1962 . , State flower : wild / native forget-me-not , adopted by the Territorial Legislature in 1917 .	Click to Use Fact
Pacific Ocean	Body	The Pacific Ocean lies to the south and southwest .	Click to Use Fact
Arctic	Geography	The climate in the extreme north of Alaska is Arctic (Köppen : ET) with long , very cold winters and short , cool summers .	Click to Use Fact
			Click for No Fact

THEM: Thats cool, I love salmon. What are their other main exports economically speaking?

YOU: The US armed forces through military bases and tourism are large parts of the economy. Alaska also has a large oil business with the Trans Alaska pipeline transporting more oil than any other crude oil pipeline.

Figure 3.5: The assistant can incorporate any number of facts into their reply to the user. Their goal is to answer the user’s immediate questions, and anticipate what information they would be most interested in.

knowledge groundings—into Curiosity by annotating the dialog acts of each message. The dialog act annotation schema is based on ISO 24617-2 (Bunt et al., 2010, 2012) with customized sub-categories for our scenario; Table 3.1 shows our schema, descriptions, and examples. The dialog acts are annotated in a separate collection using a custom annotation interface (Appendix A.2). In our case, knowledge groundings paired with dialog acts are helpful for identifying assistant policies (§3.3.1.2).

3.2.4 Validating Data Quality

We crowd-sourced conversations in two phases using ParlAI (Miller et al., 2017). In the first, pilot studies collect feedback from individual workers. Based on feedback, we create task guidelines, sample dialogs, a FAQ, tutorial videos, and qualification tests. These materials were used to train and qualify crowd-workers for the second phase. During the second, we monitor the interface usage and removed workers that ignored instructions.

To validate the quality of dialog act annotations, we use Krippendorff’s α (Krippendorff, 2004). Dialog acts are multi-class and multi-label: a message can have none, one, or multiple dialog acts (e.g., positive feedback and followup). However,

Dialog Act	Count	Description	Example
request topic	10,789	A request primarily about the topic.	I'd like to know about <u>Puerto Rico</u> .
request aspect	41,701	A request primarily about an aspect.	Could you tell me about its <i>history</i> ?
request followup	4,463	A request about mentioned concept.	Do you know more about the <u>Táinos</u> ?
request other	10,077	Requests on unmentioned concepts.	What is there to know about cuisine?
inform response	59,269	Directly answer an info request.	<u>Táinos</u> were caribbean indigenous.
inform related	6,981	Not a direct answer, but related info.	I do not know, but...
inform unrelated	557	Does not answer question, not related.	Politics is tiring!
feedback positive	26,946	Provide positive feedback	Thats quite interesting!
feedback negative	176	Provide negative feedback	Thats pretty boring.
feedback ask	36	Ask for feedback	Do you find < info > interesting?
offer topic	91	Offer to discuss topic	Want to learn about <u>Puerto Rico</u> ?
offer aspect	1,440	Offer to discuss aspect	How about more on its <i>demographics</i> ?
offer followup	63	Offer to discuss mentioned concept.	I could say more about the <u>Spanish</u> .
offer other	1,619	Offer to discuss unmentioned concept.	How about I tell you about its exports.
offer accept	1,727	Accept offer of information.	I'd love to learn about its <u>history</u> .
offer decline	405	Decline offer of information	Sorry, I'm not interested in that.

Table 3.1: Counts, descriptions and examples of the dataset’s dialog acts.

Krippendorff’s α is typically computed for single-label tasks from a table where rows represent examples, columns represent annotators, and cells indicate the singular class label. We convert our multi-label problem to a single label problem by making each combination of example and label class a row in the table (Table 3.2). Since there are few dialog acts per utterance, most annotations agree; however, since Krippendorff’s α focuses on disagreement, it is appropriate for this scenario. Using a separate interface (Appendix A.2), we doubly annotate 4,408 dialogs and the agreement score 0.834 is higher than the 0.8 threshold recommended by Krippendorff (2004). Next, we analyze the annotated dialogs and introduce our model.

3.3 Dataset Analysis

Next, we show that users prefer aspect-specific, rooted facts and then show general statistics of the Curiosity dataset.

	Annotator 1	Annotator 2
Utterance 1, Label A	Yes	No
Utterance 1, Label B	Yes	No
Utterance 2, Label A	Yes	Yes
Utterance 2, Label B	Yes	Yes

Table 3.2: Consider a task where each utterance has labels A and B. In the single-label version, each utterance is labeled as either A or B. The table shows the outcome of converting the multi-label version to single-label by creating a row for each example-label combination. Cell values are binary indicators.

3.3.1 What Facts do User Prefer?

Earlier, we hypothesized that when assistants use facts that mention previously known entities (rooted facts), that users will be more likely to engage (§3.1). In our data collection, we incorporate two mechanisms to test this hypothesis. The first mechanism is explicit: we directly ask users—through a like button—to indicate what messages they preferred. The second mechanism is implicit and derived by mining dialogs for specific sequences of dialog acts that suggest engagement with the content. For each of these mechanisms, we compute the likelihood $P(\text{Prefer} \mid \text{Fact Source})$ of a user preferring utterances grounded to each fact source (Rooted, Aspect, or General). Figure 3.6 shows this likelihood and indicates that users prefer: (1) facts relevant to aspects versus general ones and (2) rooted facts in three of four scenarios.

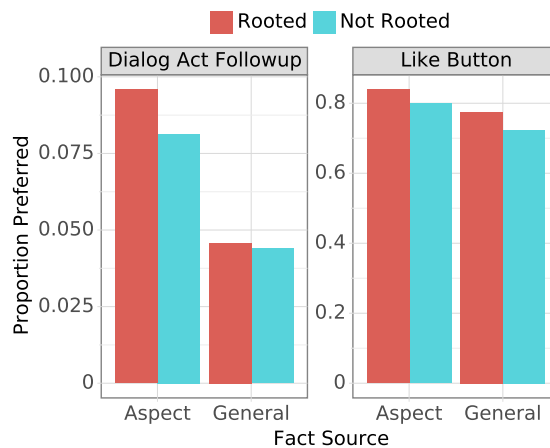


Figure 3.6: User engagement is measured by dialog act followups (left) and like button usage (right). We compare reactions to messages that use a fact mentioning an entity the user knew about (rooted) and whether the fact is general or aspect-specific. Pairwise differences are statistically significant (99%+) with a two proportion z-test except for dialog act followups between rooted and non-rooted general facts. Overall, users prefer on-aspect, rooted facts.

3.3.1.1 Likes for Explicit Preference Elicitation

Explicit preference is computed directly from like button usage and shown on the right panel of Figure 3.6. Overall, users liked 60% of messages, and they prefer on-aspect, rooted facts.

3.3.1.2 Mining Acts for Implicit Preferences

When users ask specific followup questions—as opposed to generic ones—about an assistant’s fact, it shows that the user implicitly prefers these kinds of messages. For example, asking about an entity like the Taínos is more specific than asking about history and therefore indicates engagement. We identify these interactions by mining for assistant-user message pairs where the assistant uses a fact and their message is has an “inform” dialog act. With these, we compute the likelihood

$$P(\text{Outcome} = \text{request followup} \mid \text{Fact Source})$$

that the user’s message has the “request followup” dialog act given the source. Similarly to likes, users engage more with aspect-oriented and rooted facts.

3.3.1.3 Paraphrase Analysis

Although our work does not include a paraphrase model, we manually analyze a random sample of two hundred and fifty assistant messages where facts were used (Figure 3.3). Of these messages, 51% were acceptable paraphrases, 27% were verbatim copies, 12% were contextualizations of near copies, and the remainder were errors such as incorrect paraphrases or did not incorporate the fact. Appendix A.4 shows descriptions, counts, and random examples of each category. This analysis estimates that about half of grounded messages have non-trivial signal for future paraphrase models to use.

3.3.2 Dataset Statistics

Table 3.4 shows the basic statistics of the Curiosity dataset which contains 14,048 dialogs with 181,068 utterances. The fact database contains 93,845 facts; of those, 76,120 (81%) were shown to the assistants and 27,486 (29%) were used in at least one message. We randomly split dialogs for training, validation, and testing.

3.4 Models

We design a machine learning model that predicts assistant and user actions. We introduce a multi-task architecture for Curiosity that **H**ierarchically **M**odels (CHARM, Figure 3.7) dialogs to: (1) predict the dialog acts of the user message (utterance act prediction), (2) select the best fact (fact prediction), (3) choose the best set of dialog acts for the next message (policy act prediction), and (4) predict if the assistant message will be liked (like prediction).

Category	Label	Count	Percent
Copy	verbatim	68	27.2%
Copy	cherry-pick	6	2.40%
Copy	context	30	12.0%
Copy	Total	104	41.6%
Paraphrase	paraphrase-correct	111	44.4%
Paraphrase	paraphrase-multiple	17	6.80%
Paraphrase	Total	128	51.2%
Error	paraphrase-error	5	2.00%
Unrelated	unrelated	13	5.20%
Total		250	100%

Table 3.3: We analyze the paraphrases annotators use through manual categorization. The ‘‘Copy’’ category includes cherry-picked verbatim phrases, verbatim copies, and contextualized copies (e.g., changing a named entity to ‘‘it’’). The majority of paraphrases are correct and only incorporate the provided fact, but a few weave in other information. 7.2% of paraphrases are either unrelated to the selected facts or paraphrase the fact incorrectly. Overall, 51.2% of messages have valid paraphrases.

3.4.1 Text Representation

CHARM jointly encodes the text of utterances and facts with one encoder ENC. Following our prior encoder notation (§2.4.3), ENC is a bi-directional LSTM (Sutskever et al., 2014) over concatenated GloVe (Pennington et al., 2014) word embeddings and Wikipedia2Vec (Yamada et al., 2018a) entity embeddings.⁵ The text t_i^u of utterance u_i in dialog D is represented as $\text{ENC}(t_i^u)$. Similarly, fact f_j on turn i is represented as $\text{ENC}(t_{i,j}^f)$ where j indexes facts shown on that turn.

3.4.2 Dialog Representation

In our models, we use a hierarchical recurrent encoder (HRE) architecture (Sordani et al., 2015; Serban et al., 2015) where a forward LSTM contextualizes each utterance to the full dialog. We modify the HRE model by adding additional inputs beyond the utterance’s textual representation. First, we represent user’s known entities

$$\mathbf{k} = \text{avg}(\text{ENC}_{\text{entity}}(e_1), \dots, \text{ENC}_{\text{entity}}(e_k)) \quad (3.1)$$

as the average of entity embeddings. An entity embedding also represents the topic

$$\mathbf{t} = \text{ENC}_{\text{entity}}(\text{topic}) \quad (3.2)$$

⁵In CHARM, BERT was not as effective an encoder.

Metric (# of)	Total	Train	Val	Test	Zero
Dialogues	14,048	10,287	1,287	1,287	1,187
Utterances	181,068	131,394	17,186	17,187	15,301
Likes	57,607	41,015	5,928	5,846	4,818
Topics	361	331	318	316	30
Facts Total	93,845	NA	NA	NA	NA
Facts Shown	76,120	66,913	29,785	30,162	6,043
Facts Used	27,486	21,669	4,950	4,952	2,290

Table 3.4: Curiosity has 14,048 dialogs. On average, dialogs have 12.9 utterances. 60% of the assistants’ 90,534 utterances were liked.

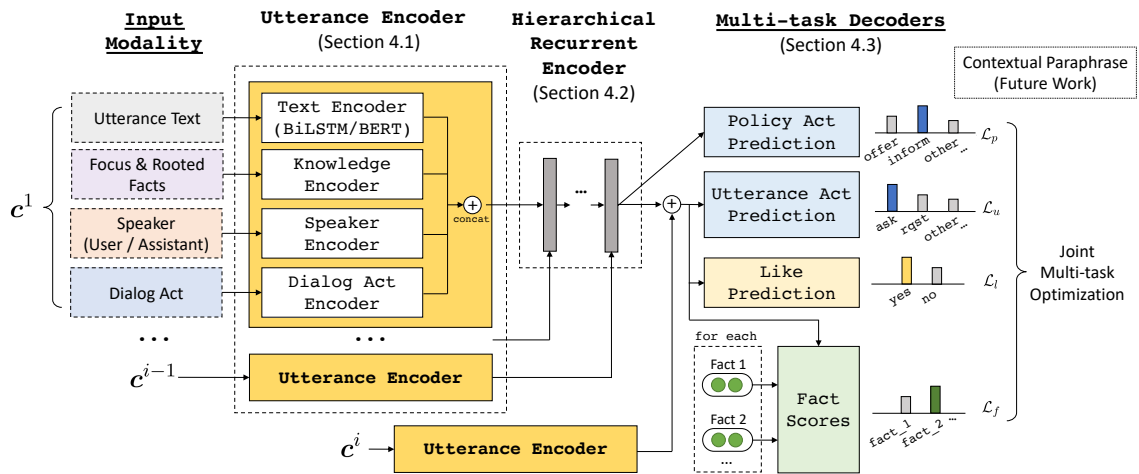


Figure 3.7: **Architecture**: CHARM builds a dialog context up to $t = i - 1$ to predict the current message’s dialog acts (policy prediction) and the best facts to use. The model uses this combined with the current utterance to classify it’s dialog acts and if it will be liked.

of the dialog. Next, we create trained speaker embedding v_s for the user and v_t for the assistant. Given the set of all dialog acts \mathcal{A} , each utterance has a set of dialog acts $\mathcal{A}_u \in \mathcal{P}(\mathcal{A})$ where $\mathcal{P}(\mathcal{X})$ denotes the set of all subsets of \mathcal{X} . Finally, we use an act embedder ACT to compute an act representation

$$\mathbf{a}^{(i)} = \frac{1}{|\mathcal{A}_u|} \sum_{a_k \in \mathcal{A}_u} \text{ACT}(a_k) \quad (3.3)$$

by averaging embeddings at each turn. The input at each step is the concatenation

$$\mathbf{c}^{(i)} = [\text{ENC}(t_i^u); \mathbf{a}^{(i)}; \mathbf{t}; \mathbf{k}; \mathbf{v}] \quad (3.4)$$

of the representations for text, speaker, topic, known entities, and utterance dialog acts.⁶ With this joint representation, the contextualized dialog representation

$$\mathbf{h}^{(i-1)} = \text{LSTM}(\mathbf{c}^{(1)}, \dots, \mathbf{c}^{(i-1)}) \quad (3.5)$$

⁶The speaker embedding v alternates between v_s and v_t .

is the final LSTM state and includes time step $t = i - 1$. The dialog up to and including time i is

$$\mathbf{d}^{(i)} = [\mathbf{h}^{(i-1)}; \mathbf{c}^{(i)}] \quad (3.6)$$

which emphasizes the current utterance and makes multi-task training straightforward to implement.

3.4.3 Tasks and Loss Functions

In our model, we jointly learn to predict fact usage, user likes, utterance acts, and policy acts.

Fact Prediction For every assistant turn, the model predicts which fact(s) from

$$\{f_1, \dots, f_k\} \in \mathcal{F}^{(i)}, \mathcal{F}^{(i)} \in \mathcal{P}(\mathcal{F})$$

the assistant marked as “used” where \mathcal{F} is the set of all facts. We frame this task as pointwise learning to rank (Li et al., 2008). A fact prediction network

$$\mathbf{s}_j^{f,(i)} = \text{GELU} \left(\left[\mathbf{W}^f \cdot \mathbf{h}^{(i-1)} + \mathbf{b}^f; \text{ENC}(t_j^f) \right] \right) \quad (3.7)$$

with parameters \mathbf{W}^f and \mathbf{b}^f using a Gaussian Error Linear Unit (Hendrycks and Gimpel, 2017b) outputs salience scores for each fact. The network does not use utterance u_i since it contains signal from the choice of fact. The predictions

$$\hat{\mathbf{y}}_j^{f,(i)} = \text{softmax}(\mathbf{s}_j^{f,(i)}) \quad (3.8)$$

are converted to probabilities by the softmax

$$\text{softmax}(\mathbf{q}) = \frac{\exp(\mathbf{q})}{\sum_{j=1}^k \exp(\mathbf{q}_j)} \quad (3.9)$$

over k labels. Using this, we compute the fact loss

$$\mathcal{L}_f = \frac{1}{|\mathcal{F}^{(i)}|} \sum_{i,j} \ell_{ce}(\hat{\mathbf{y}}_{i,j}^f, \mathbf{y}_{i,j}) \quad (3.10)$$

where labels $\mathbf{y}_j^{f,(i)}$ indicate if fact from utterance i in position j was used and

$$\ell_{ce}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{p=1}^k \mathbf{y}_p \log(\hat{\mathbf{y}}_p). \quad (3.11)$$

is the cross entropy loss. To mitigate class imbalance, we also scale positive classes by nine (Japkowicz and Stephen, 2002).

Policy Act and Utterance Act Prediction Each utterance may have multiple dialog acts so we treat policy and utterance act prediction as a multi-label task. The goal of policy prediction is to choose the best act for the next utterance; the utterance act classifies the last message’s acts. To predict these acts, we create a policy act network

$$\mathbf{s}^{p,(i)} = \text{GELU}(\mathbf{W}^p \cdot \mathbf{h}^{(i-1)} + \mathbf{b}^p) \quad (3.12)$$

and an utterance act network

$$\mathbf{s}^{u,(i)} = \text{GELU}(\mathbf{W}^u \cdot \mathbf{d}^{(i)} + \mathbf{b}^u) \quad (3.13)$$

where the probability of act a_k is $p_k^{*,i} = \exp(\mathbf{s}_k^{*,(i)})$. From these, we derive the policy act loss

$$\mathcal{L}_p = \sum_k^{|A|} y_{i,k}^a \log p_k^{p,(i)} + (1 - y_{i,k}^a) \log(1 - p_k^{p,(i)}) \quad (3.14)$$

and utterance act loss

$$\mathcal{L}_u = \sum_k^{|A|} y_{(i),k}^a \log p_k^{u,(i)} + (1 - y_{(i),k}^a) \log(1 - p_k^{u,(i)}) \quad (3.15)$$

for an utterance at $t = i$ with act labels $y_{i,k}^a$.

Like Prediction For every assistant message, the model predicts the likelihood of the user “liking” the message. We treat this as binary classification, predict the “like” likelihood

$$\hat{y}_i^l = \text{softmax}(\text{GELU}(\mathbf{W}^l \cdot \mathbf{h}^{(i)} + \mathbf{b}^l)), \quad (3.16)$$

and use it to compute the like loss

$$\mathcal{L}_l = \ell_{ce}(\hat{y}_i^l, y_i^l) \quad (3.17)$$

where y_i^l indicates if the message was liked. We train the model jointly and optimize the loss

$$\mathcal{L} = \mathcal{L}_f + \mathcal{L}_l + \mathcal{L}_p + \mathcal{L}_u. \quad (3.18)$$

3.4.4 Model Implementation and Training Details

We implement all models with PyTorch (Paszke et al., 2019) and AllenNLP (Gardner et al., 2018). The learning rates for models is set using the built-in learning rate finder in AllenNLP. Model losses were optimized with Adam (Kingma and Ba, 2015); the BERT model uses a learning rate of .0001 and CHARM a learning rate of .001 with otherwise default parameters. We train for a maximum of forty epochs and early stop based on the sum of validation losses. The CHARM model uses batch size 64 and the BERT model batch size 4. In our models, text encoders for utterances and facts share parameters. Additional hardware and runtime details in Appendix A.6.

Model	Fact Rank (MRR)		Utt. Act (F_1)		Policy Act (F_1)		Like (Accuracy)	
	Val	Test	Val	Test	Val	Test	Val	Test
Majority Class	N/A	N/A	0.602	0.604	0.491	0.494	0.690	0.681
E2E BERT	0.420	0.418	0.794	0.795	0.635	0.631	0.829	0.822
CHARM	0.546	0.546	0.845	0.847	0.682	0.682	0.826	0.815
– context	0.516	0.506	0.838	0.842	0.664	0.664	0.824	0.820

Table 3.5: The CHARM model outperforms end-to-end BERT on most tasks. We compare fact selection with MRR, dialog act prediction with micro-averaged F_1 , and like prediction with accuracy. Ablating dialog history degrades context-dependent tasks (fact selection and policy act prediction), but not tasks more dependent on one message.

3.5 Modeling Experiments

CHARM improves over a BERT model in most tasks. We evaluate each sub-task with separate metrics. Fact selection is evaluated with mean reciprocal rank (MRR). For utterances with at least one selected fact, we compute the MRR using the selected facts as relevant documents. We compare like prediction with binary classification accuracy. For utterance and policy act prediction, we compare models with micro-averaged F_1 scores so that frequent classes are weighted more heavily. For each metric, we report validation and test set scores.

3.5.1 Baselines

BERT (Devlin et al., 2018) is a standard baseline for many NLP tasks. We use a multi-task extension of an uncased BERT model as our primary baseline and fine-tune it for our unique set of tasks (E2E BERT). Specifically, we use the CLS representation of each utterance to replace the HRE representation as a time-distributed input to the same multi-task decoders (Section 3.4.3). The context-less CHARM ablation replaces the dialog contextualizer LSTM with a per-timestep projection layer. Lastly, we report majority class accuracy for classification tasks.

3.5.2 Discussion

The proposed CHARM model for conversational curiosity is more effective than E2E BERT for most of the tasks in Curiosity (Table 3.5). Specifically, CHARM improves significantly in fact prediction (13 MRR points) and both dialog act prediction tasks (5 F_1 points), demonstrating the efficacy of the structural encoding of the various input modalities. Generally, models accurately predict utterance acts and likes, but their MRR and F_1 scores on fact selection and policy act prediction is comparatively worse. To a degree, this is expected since there is not always one best fact or one best action to take as the assistant; there may be various reasonable choices, which is common in information retrieval tasks. Nonetheless, models that specif-

ically factor in the relationship between prior knowledge and entities should yield improvements. For example, [Liu et al. \(2018\)](#) predict the most relevant unmentioned entity while [Lian et al. \(2019\)](#) model a posterior distribution over knowledge. We leave these improvements to future work.

3.6 Related Work

Our work builds on knowledge-grounded conversational datasets and modeling.

Datasets Although there are numerous grounded datasets, we did not find one for conversational information seeking that contained fine-grained knowledge groundings, message-level feedback from the user, and dialog acts. Table 3.6 compares the Curiosity dataset to several others according to six factors: (1) is the goal of the task information seeking, (2) is the dataset collected from natural dialog with one participant always taking the role of an assistant, (3) are dialog responses constrained, (4) are document groundings annotated—as opposed to distantly supervised—and fine-grained, (5) is there message level feedback for the assistant, and (6) is the dataset annotated with dialog acts.

Our dataset is most similar to those for information-seeking such as QuAC ([Choi et al., 2018](#)), Wizard of Wikipedia ([Dinan et al., 2019b](#), WoW), CMU DOG ([Zhou et al., 2018b](#)), MS MARCO ([Nguyen et al., 2016](#)), Topical Chat ([Gopalakrishnan et al., 2019](#)), the TREC Conversational Assistance track ([Dalton et al., 2019](#), CASt), and Search as a Conversation ([Ren et al., 2020](#), SaaC). QuAC constrains assistant responses to spans from Wikipedia, which makes it better for conversational question answering, but prevents more sophisticated assistant policies. QuAC also provides dialog acts, but they exist so that the assistant can inform the user of valid actions; we annotate dialog acts after-the-fact so that we can compare freely chosen user responses. Like QuAC, Topical Chat, SaaC, and WoW have annotated knowledge-groundings for each message, but responses are free-form. SaaC is a contemporaneous, CASt-based dataset that shares our motivation to make conversation a medium for information-seeking. Topical Chat includes user feedback, but instead of explicitly defined roles, workers implicitly take dual and alternating roles as the user and assistant through knowledge asymmetry; followup work added automatically annotated dialog acts to Topical Chat ([Hedayatnia et al., 2020](#)).

Many tasks instruct annotators to take on a specific role in the dialog. For example, in Wizard of Wikipedia, annotators assume an assigned persona ([Zhang et al., 2018a](#)) in addition to being the user or assistant. Consequently, many dialogs revolve around personal discussions instead of teaching about a topic. Additionally, annotators may not have the background to play their role. In contrast, we ask annotators to take roles that—as humans—they already know how to do: read about and convey interesting information on a topic (assistant) and engage in inquiry about a novel topic (user).

Our work is one of many in knowledge-grounded conversational datasets. For example, [Moghe et al. \(2018\)](#) have workers discuss movies and ground messages to

Dataset	Info Seek- ing	Dialog w/Assistant	Free Re- sponse	Annotated Ground- ing	Message Feed- back	Dialog Acts
Curiosity (ours)	✓	✓	✓	✓	✓	✓
Topical Chat (Gopalakrishnan et al., 2019)	✓	⚠	✓	✓	✓	⚠
Search as a Conversation (Ren et al., 2020)	✓	✓	✓	✓	✗	✗
Wizard of Wikipedia (Dinan et al., 2019b)	✓	✓	✓	✓	✗	✗
QuAC (Choi et al., 2018)	✓	✓	✗	✓	✗	⚠
CMU DOG (Zhou et al., 2018b)	✓	✓	✓	⚠	✗	✗
MS Marco Conversational (Nguyen et al., 2016)	✓	✗	N/A	N/A	N/A	N/A
OpenDialKG (Moon et al., 2019)	✗	✓	✓	✓	✗	✗
CoQA (Reddy et al., 2019)	✗	✓	⚠	✓	✗	✗
Holl-E (Moghe et al., 2018)	✗	⚠	✓	✓	✗	✗
Commonsense (Zhou et al., 2018a)	✗	✗	✓	✗	✗	✗
Reddit+Wiki (Qin et al., 2019)	✗	✗	✓	✗	✗	✗

Table 3.6: ✓ indicates a dataset has the feature, ⚠ that it does with a caveat, and ✗ that it does not. Conversational MS MARCO is a search dataset but has inquiry chains we want assistants to induce (exemplar in Appendix A.7). Topical Chat and Search as a Conversation are motivationally similar. While our dataset’s combination of (human) annotation is unique, all three datasets are steps forward in resources for conversational information-seeking.

plot descriptions, reviews, comments, and factoids; however, one worker plays both roles. In OpenDialKG (Moon et al., 2019), annotators ground messages by path-finding through Freebase (Bast et al., 2014) while discussing and recommending movies, books, sports, and music. Qin et al. (2019) use Reddit discussion threads as conversations and ground to web pages. Similarly, Ghazvininejad et al. (2018) collect Twitter three-turn threads and ground to Foursquare restaurant reviews. Our work adds to this dataset compendium.

External Knowledge in Models Our model is related to those that incorporate external information like facts in question answering (Weston et al., 2015; Sukhbaatar et al., 2015; Miller et al., 2016a), knowledge base triples in dialog models (Han et al., 2015; He et al., 2017; Parthasarathi and Pineau, 2018), common sense (Young et al., 2017; Zhou et al., 2018a), or task-specific knowledge (Eric and

Manning, 2017). Similarly to Kalchbrenner and Blunsom (2013); Khanpour et al. (2016), CHARM predicts the act of the current message, but also next message’s act like Tanaka et al. (2019) do.

3.7 Future Work and Conclusion

This chapter primarily introduces Curiosity: a large-scale conversational information-seeking dataset. Like much work in IR and the Cranfield paradigm, the primary goal of this dataset is to help build models that better satisfy users—in this case, in the context of curiosity-driven dialog. Aside from building the Curiosity dataset, this work also investigates an alternative, dialog act based method for measuring how often users meaningfully engage with assistant messages which we take as a proxy for their satisfaction of how interesting the assistant is. With Curiosity’s unique set of annotations, we design CHARM which jointly learns to choose facts, predict a policy for the next message, classify dialog acts of messages, and predict if a message will be liked. We hope that the Curiosity dataset and contemporaries will encourage progress and interest in curiosity-driven dialog.

Since we later discuss avenues for more ambitious future work (Chapter 7), here we discuss three immediate directions for work using the Curiosity dataset. The first is to augment our CHARM model with a text generation module to make a digital version of our human assistants. This involves contextualizing and paraphrasing facts which our dataset supports. Second, dialog act sequences could identify additional data-driven policies that could be used to define rewards or losses. By conditioning on dialog acts or sequences of dialog acts, textual outputs could be better-controlled (Sankar and Ravi, 2019; See et al., 2019) and combined with knowledge grounding (Hedayatnia et al., 2020). However, text is not the native modality of digital assistants.

We envision digital assistants participating in information-seeking, which means *also* handling speech input. Consequently, automatic speech recognition (ASR) introduces transcription errors which are especially prevalent in knowledge-oriented text like question answering (Peskov et al., 2019). Gopalakrishnan et al. (2020) show this is also problematic in information-seeking dialog by comparing models on textual and ASR versions of Topical Chat. To close the loop in conversational information-seeking, models need to account for the speech-based environment of digital assistants.

Lastly, for the focus we give to dialog act mining, there is room for significant room for improvement in the evaluation of models for predicting these. There are two drawbacks to the current evaluation approach. First, although the dialog act of interest to us is relatively rare (“request followup”), the dialog act evaluations use micro-averaged F1 scores which discounts the importance of these rare classes.⁷ Second, these rare classes are more challenging to detect than some of the most common classes like “request topic.” That is, a priori, we expect that the most

⁷At best, macro-averaging F1 would make the class weights equal, but also scale by the number of distinct classes.

frequent classes are not only the easiest for a model to predict,⁸ but are the ones of least interest to us. Ideally, we would like to have an evaluation methodology that infers which types of examples are difficult and rewards models appropriately. Having seen the Curiosity dataset as an example of the Cranfield paradigm, the next chapter introduces a solution to this particular challenge—which is one shared across many QA tasks.

⁸Even ignoring that this class has more training data, we still maintain it is intrinsically more difficult to detect.

Chapter 4: Evaluation Examples are not Equally Informative: How should that change NLP Leaderboards?

A successful evaluation requires a task that is neither too easy nor too difficult for the current technology. If the task is too simple, all systems do very well and nothing is learned. Similarly, if the task is too difficult, all systems do very poorly and again nothing is learned.

Ellen M. Voorhees, The TREC-8 QA Track Report

Before shifting focus to a Manchester paradigm task (Chapter 5), we first turn our attention to the challenge of evaluating tasks for which the test examples are not equally difficult.¹ In Chapter 3, one example of this problem is evaluating how well a dialog act classifier works since the most difficult class factors least into aggregate metrics since it is also a rare class. Taking a wider view, the main problem is that standard evaluation schemes weigh examples equally, but a cursory examination shows this assumption to be flawed. Intuitively, any student could tell you that exam questions are not equally difficult; this is even trivially verifiable in NLP datasets: poorly annotated examples—which are present to some degree in many datasets—tend to be either too difficult or too easy. This applies equally well to most QA evaluations since they use metrics like mean accuracy or mean lexical overlap that weigh each question equally.

Rather than re-invent the wheel, to address this problem we take inspiration from work in a field that concerns itself with faithfully and reliably measuring skills using exams—educational testing. Ultimately, this chapter (1) introduces a method for weighting examples appropriately and (2) argues we should maximize the ability of QA evaluations to discriminate between better and worse models. Chapters 5 and 6 subsequently introduce methods for improving the discriminative power of QA evaluations through better formats and data. To test our method, this chapter takes a critical look at the dominant evaluation methodology in QA: the leaderboard.

4.1 Leaderboards are Shiny

Leaderboard evaluations—for better or worse—are the *de facto* standard for measuring progress in question answering (Rajpurkar et al., 2016) and in many NLP

¹This chapter is based on the ACL publication Rodriguez et al. (2021).

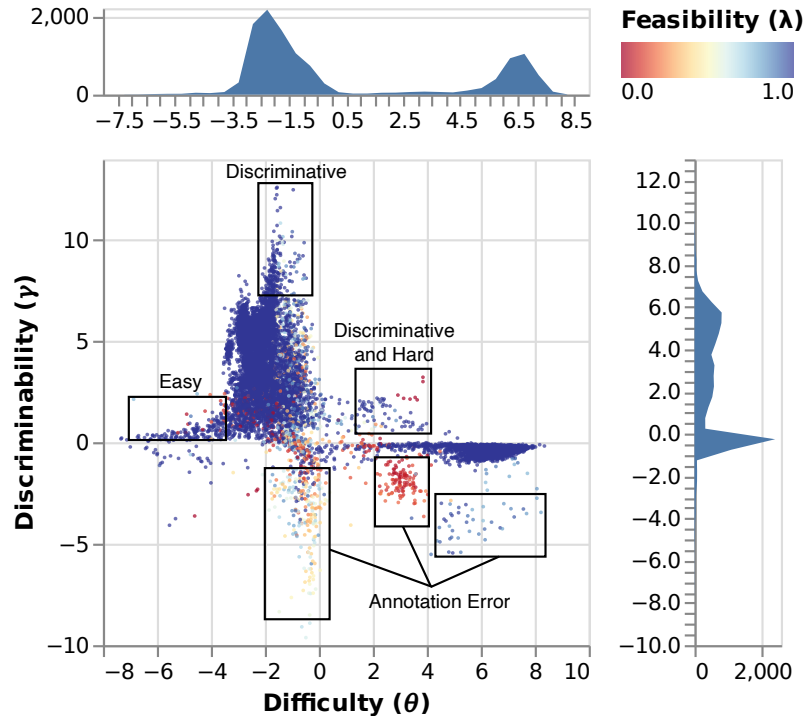


Figure 4.1: Difficulty and Ability Discriminating (DAD) leaderboards infer the difficulty, discriminativeness, and feasibility of examples. Negative discriminability suggests an annotation error; for example, the question with most negative discriminability asks “Why did demand for rentals decrease?” when the answer is “demand for higher quality housing increased.”

tasks (Wang et al., 2019a). An unfortunate side effect of leaderboard popularity is SOTA-chasing, often at the expense of carefully inspecting data and models (Linzen, 2020). For example, the same models that achieve “super-human” question answering (Najberg, 2018) often fail spectacularly (Feng et al., 2018; Wallace et al., 2019a) by learning non-generalizable statistical patterns (McCoy et al., 2019; Niven and Kao, 2019); i.e., these are the models that Levesque (2014) refers to as “idiot-savants” (§1.1). Finally, focusing *solely* on metrics conflates progress on a specific *task* with progress on real-world NLP *problems* that the task stands in for (Bender and Koller, 2020). Plainly, focusing on headline SOTA numbers “provide(s) limited value for scientific progress absent insight into what drives them” and where they fail (Lipton and Steinhardt, 2019).

In this chapter we take leaderboards “as they are,” and imagine how they might better support research. Leaderboards establish differences between models on a fixed task. Hence, leaderboards should enable and encourage the comparison of models and inspection of examples. And leaderboards should also reveal when they have outlived their usefulness (Boyd-Graber and Börschinger, 2020).

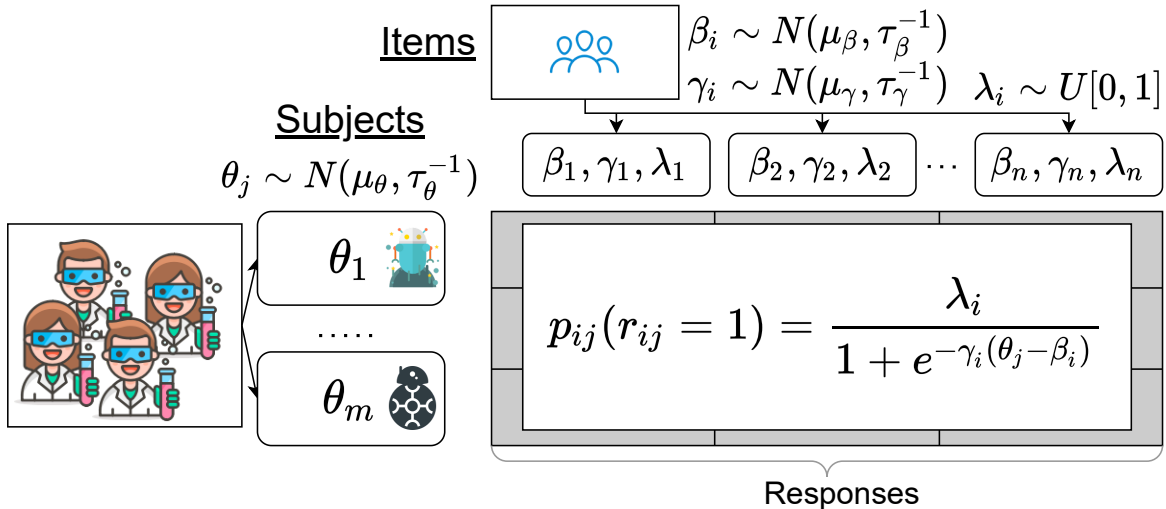


Figure 4.2: A DAD leaderboard uses IRT to jointly infer item difficulty β_i , discriminability γ_i , feasibility λ_i , and subject skill θ_j . These predict the likelihood $p_{ij}(r_{ij} = 1)$ of a correct response r_{ij} .

4.1.1 How to Direct Leaderboards’ Light

To help focus attention on examples and models of interest, we propose Difficulty and Ability Discriminating (DAD) leaderboards that explicitly model both task and submissions *jointly*, rather than either in isolation. DAD’s underlying model is based on Item Response Theory (Lord et al., 1968; Baker, 2001, IRT, reviewed in §4.2), a widely used (van Rijn et al., 2016) alternative in educational testing to simple summary statistics (Edgeworth, 1888).

DAD can explicitly identify the *difficulty* and *discriminability* of items (Figure 4.1),² which in turn can lead to a more nuanced ranking of models, identifying poor items, and better understanding of a dataset and task. Throughout this chapter, we use the question answering (QA) benchmark SQuAD 2.0 (Rajpurkar et al., 2018). For example, DAD can identify questions that are challenging to models and questions that are wrong (incorrectly annotated). In addition to better understanding datasets, it is also helpful for efficiently selecting evaluation to annotate. We conclude the chapter with recommendations for future leaderboards (§4.7) and discuss where IRT in NLP can go next (§4.9).

4.2 A Generative Story for Leaderboards

Much as questions are the product of their inputs (§2.2.3), leaderboards are a product of the metrics, evaluation data, and subjects (machine or human) who answer items (Figure 4.2). For concreteness, let’s assume that we have a question answering task and two subjects: Ken, who is good at trivia, and Burt, who is not. In the simplest IRT models, each subject j has a random variable θ_j corresponding to their skill: Ken’s is big, Burt’s is small.

²Example and feasibility distribution in Appendix B.1.

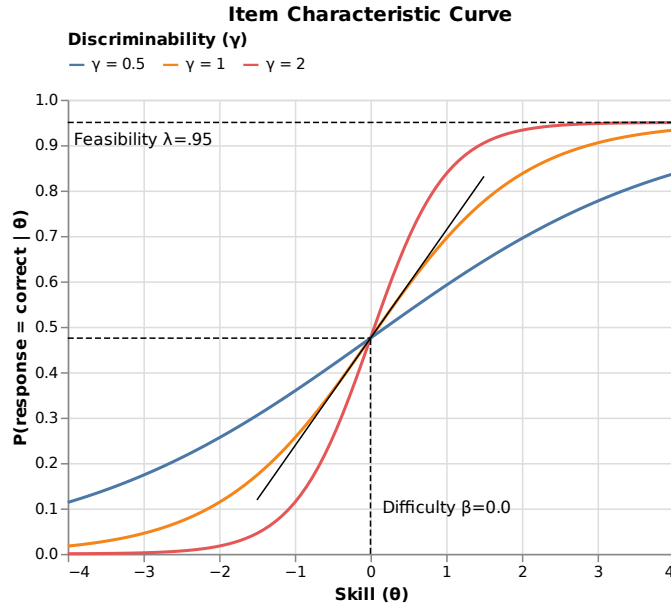


Figure 4.3: In IRT, the Item Characteristic Curve describes the probability that a specific item with difficulty β will be answered correctly as a function of skill θ . Visualizing the parameters is helpful in a few ways. First, it shows that high discriminability leads to larger differences in the maximal and minimal correctness probability (for a given skill range), and the tangent line visually links discriminability to the slope. Second, it clearly shows the maximal probability when feasibility is greater than zero. Lastly, it demonstrates that the inflection point is at the point where difficulty and skill equal out (in this case, zero).

But you cannot know that until you start asking them questions of varying *difficulty* β_i . Harder questions have a higher difficulty (“what is the airspeed of an unladen swallow”) than easy ones (“who is buried in Grant’s tomb”). The bigger the margin between a subject’s skill θ_j and an item’s difficulty β_i , $\theta_j - \beta_i$, the more likely that subject j responds correctly $p_{i,j}(r_{i,j} = 1)$. This is the simplest IRT model, which we call **IRT-base**.

Generally, given n test items $\mathcal{X} = (X_1, \dots, X_n)$ and m subjects $\mathcal{S} = (S_1, \dots, S_m)$, where each subject responds to every item, we want to estimate subject skills and item difficulties. To discover the random values that best explain the data, we turn to probabilistic inference (Pearl, 1988).

Two additional random variables further improve DAD: discriminability γ_i and feasibility λ_i . Let’s return to the margin between a question’s difficulty β_i and a subject’s skill θ_j . A discriminative question is challenging but can still be answered correctly by a strong subject. If Ken’s ability is higher than most items’ difficulty ($\theta_j - \beta_i$ is large), item discriminability multiplies this gap by γ_i in a model called **IRT-disc**. Questions with low γ_i (discriminability) are low quality: they have annotation error or do not make sense.

Another way of capturing poor quality questions is the feasibility λ_i . For example, if the question “who was the first president” has the answer **Rajendra**

Prasad, the question has an unstated implicit assumption that subjects must guess what country or company the question is about. In the **IRT-feas** model (Figure 4.3), if a large fraction of subjects all get an item wrong, everyone’s probability of getting the item right is capped at feasibility λ_i . In NLP terms, $1 - \lambda_i$ corresponds to the prevalence of annotation errors that lead to unsolvable items. Having introduced the constituent elements of the model, we now present the full generative model:

1. For each subject j :
 - (a) Draw skill $\theta_j \sim \mathcal{N}(\mu_\theta, \tau_\theta^{-1})$
2. For each item i :
 - (a) Draw difficulty $\beta_i \sim \mathcal{N}(\mu_\beta, \tau_\beta^{-1})$
 - (b) Draw discriminability $\gamma_i \sim \mathcal{N}(\mu_\gamma, \tau_\gamma^{-1})$
 - (c) Draw feasibility $\lambda_i \sim \text{U}[0, 1]$
3. Draw subject i response on item j ,

$$r_{ij} \sim p_{ij}(r_{ij} \mid \theta_j, \beta_i, \lambda_i) = p_{ij}(r_{ij} = 1 \mid \theta_j) = \frac{\lambda_i}{1 + e^{-\gamma_i(\theta_j - \beta_i)}}. \quad (4.1)$$

For IRT-base, γ_i and λ_i are fixed to 1.0, while for IRT-disc, only λ_i is fixed.³

Means $\mu_\theta, \mu_\beta, \mu_\gamma$ are drawn from $\mathcal{N}(0, 10^6)$ and $\tau_\theta, \tau_\beta, \tau_\gamma$ from a $\Gamma(1, 1)$ prior, as in Lalor et al. (2019) and recommended by Natesan et al. (2016).⁴

Because it is difficult to completely codify skill and difficulty into a single number, we can rewrite the exponent in Equation 4.1 as a sum over dimensions $-\gamma_i(\sum_k \theta_{j,k} - \beta_{i,k})$ where each dimension captures the interaction between an item’s difficulty and a subject’s skill. For example, perhaps Burt could better exploit artifacts in one dimension (their skill for $\theta_{j,k=5}$ is high but everywhere else is low) while Ken might not know much about a particular topic like potent potables ($\theta_{j,k=2}$ is low but everywhere else is high). We call this model **IRT-vec**.⁵ Multidimensional IRT models (Reckase, 2009) could—in addition to better modeling difficulty—also cluster items for interpretation; we briefly experiment with this (Appendix B.6), but leave more to future work (§4.9).

4.2.1 Examples are Not Equally Useful

IRT’s fundamental assumption is that not all items and subjects are equal. This explains why leaderboards can fail while having “normal looking” accuracies. As a thought experiment, consider a dataset that is one third easy ($\beta_i \in [0, 1]$), one third medium difficulty ($\beta_i \in [2, 3]$), and one third hard ($\beta_i \in [6, 7]$). Suppose that Ken has skill $\theta_k = 4$ while Burt has skill $\theta_b = 2$. A standard leaderboard would say

³In psychometrics, IRT-base is called a Rasch (Rasch, 1960) or 1 parameter logistic (1PL) model, IRT-disc is a 2PL model, and IRT-feas is a 4PL model with guessing set to zero.

⁴We differ by allowing $\gamma < 0$ to help identify bad items.

⁵We do not incorporate feasibility into the IRT-vec model since it already improves over 1D models without it.

that Ken has higher accuracy than Burt. But suppose there’s a new subject that wants to challenge Ken; they are not going to reliably dethrone Ken until their skill θ_c is greater than six.

This is a more mathematical formulation of “easy” and “hard” dataset splits in question answering (Sugawara et al., 2018; Rondeau and Hazen, 2018; Sen and Saffari, 2020). In IRT-feas, this recapitulates the observation of Boyd-Graber and Börschinger (2020) that annotation error can hinder effective leaderboards. IRT helps systematize these observations and diagnose dataset issues.

4.2.2 Inference

To estimate the latent parameters of our model, we use mean-field variational inference (Jordan et al., 1999). In variational inference, we propose a distribution over the latent variables, $q_\phi(\cdot)$, that approximates the true but intractable posterior $p(\cdot)$. We then minimize the KL-divergence between these distributions, equivalent to maximizing the evidence lower-bound (ELBO) with respect to the variational parameters.

In our case, $q_\phi(\cdot)$ is a mean-field distribution, which means it factorizes over each of the latent variables (the product is over the $n \times m$ subject-item pairs)

$$q_\phi(\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mu}, \boldsymbol{\tau}) = q(\boldsymbol{\mu})q(\boldsymbol{\tau}) \prod_{i,j} q(\theta_j)q(\beta_i)q(\gamma_i)$$

Specifically, for our key latent variables $z \in \{\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\gamma}\}$, the associated variational distributions are of the form $q(z) = \mathcal{N}(u_z, t_z^{-1})$. Recall that in the generative distribution, each latent z is drawn from a $\mathcal{N}(\mu_z, \tau_z^{-1})$ whose parameters are also latent variables; for these variables, we use the variational distributions $q(\mu_z) = \mathcal{N}(u_{\mu_z}, t_{\mu_z}^{-1})$ and $q(\tau_z) = \Gamma(a_{\tau_z}, b_{\tau_z})$. We optimize the ELBO with respect to the variational parameters

$$\boldsymbol{\phi} = \{\mathbf{u}_z, \mathbf{t}_z, \mathbf{u}_{\mu_z}, \mathbf{t}_{\mu_z}, \mathbf{a}_{\tau_z}, \mathbf{b}_{\tau_z}, \boldsymbol{\lambda}\}$$

for all z using ADAM (Kingma and Ba, 2015).

With DAD’s leaderboard IRT model introduced, we next discuss how leaderboard subjects are statistically compared and alternative methods—such as using IRT parameters—to evaluate whether two models are truly different.

4.3 Ranking and Comparing Subjects

Fundamentally, the objective of comparative evaluations like leaderboards is to decide whether model A is better than model B . A thread of NLP has rightfully advocated for adding rigour to these decisions using statistics (Traub, 1997, Classical Testing Theory) where the objective is to infer a true score T from the observed test score $X = T + E$ given a measurement error E , uniform across subjects. However, in educational testing—a field measuring skill and knowledge in humans—IRT is a primary measurement instrument (Hambleton, 1991, p. 2). A major motivation for

IRT is that subjects of different skill have *different* errors. IRT explicitly accounts for the bandwidth-fidelity dilemma (McBride, 1976): items can either accurately measure a narrow ability range (fidelity) *or* inaccurately measure large ability ranges (bandwidth).⁶ This section and the next contrast methods for identifying the best model and advocate for IRT.

Implicit in nearly all leaderboard evaluations is ranking of models based on a statistic such as the average accuracy. As we show in §4.4, naïve rankings are noisier than IRT rankings.

4.4 IRT for Leaderboards

Leaderboards should do two things: (1) reliably and efficiently rank better models ahead of worse models (Tague-Sutcliffe, 1992; Voorhees, 2003a) and (2) guide inspection of items and subjects (§4.5). The first ameliorates the unavoidable randomness of finite evaluations while the second enables error analysis (Wu et al., 2019) and model probing (Belinkov and Glass, 2019; Zhang et al., 2019). First we verify that IRT models accurately predict the responses of subjects (§4.4.2). Next, a ranking stability analysis shows that IRT has modestly better reliability than classical rankings (§4.3). Lastly, using IRT to actively sample items for annotation yields rankings with better correlation to the test data (§4.4.5).

4.4.1 Why a Linear Model Baseline

At first blush, the differences between IRT and logistic regression are minimal, but we include the comparison to address natural questions from the NLP community: (1) do the idiosyncrasies of the IRT formulation hurt accuracy? (2) should we add features to better understand phenomena in the questions? (3) why not use deep models?

The next section argues that both IRT and logistic regression are accurate, even without laboriously engineered task-specific features. Adding obvious features such as item words (e.g., question) only minimally improves accuracy. We explicitly omit less interpretable deep models since we want to make leaderboards *more* interpretable.

4.4.2 Response Prediction is Accurate

Just as educational testing researchers validate IRT fit seeing if they predict subject responses correctly (American Educational Research Association, 2014), we validate how well DAD predicts whether SQuAD models get questions right.

We compare against a logistic regression linear model (LM) implemented with Vowpal Wabbit (Agarwal et al., 2014). Since integrating hand-crafted features is easy, we incorporate features derived from subject IDs; item IDs; functions of the SQuAD question, answer, and title; and IRT parameters (details in Appendix B.2).

⁶Estimation error of θ varies by position (§B.5).

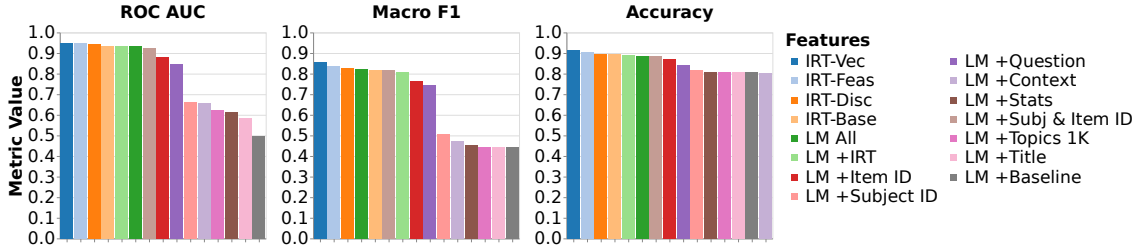


Figure 4.4: We compare each IRT and linear model (LM) by how well they predict subject responses. We focus on ROC AUC since predicting responses is an imbalanced classification problem (most subjects are correct). Under that metric, all IRT models improve over the best LM, and the strongest LM ablation only uses IRT features. That textual features are predictive in the LM suggests they could improve future models.

As in IRT, logistic regression predicts whether a subject correctly responds to an item. Later, we discuss ways to integrate more features into IRT (§4.9).

4.4.2.1 SQuAD Leaderboard Data

Experiments use data from the SQuAD 2.0 leaderboard. Development data is publicly available and test set responses were obtained from the organizers. There are 161 development subjects, 115 test subjects, and 11,873 items (1.9 million total pairs). Experiments that do not need the test responses use all the development subjects; those that do use the smaller test subset.

4.4.2.2 Evaluation Scheme

Following prior work (Wu et al., 2020), we evaluate IRT and linear models by holding out 10% of responses and computing classification metrics.⁷ In SQuAD, predicting whether a response is correct is an imbalanced classification problem (80.4% of responses in the development set are correct). Thus, we use ROC AUC, macro F1, and accuracy.

4.4.2.3 IRT Response Prediction is Accurate

IRT models that incorporate more priors into the generative story should be better, but are they? We compare four IRT models: IRT-base, IRT-disc, IRT-feas, and IRT-vec (§4.2). The more sophisticated models are better and all improve over the LM (Figure 4.4) and correlate well with other (Appendix B.3). To be clear, while outperforming the LM is good, our goal was to validate that IRT model are accurate; later, we inspect model errors and identify annotation errors (§4.5).

⁷Everywhere else in the chapter, we train on all responses.

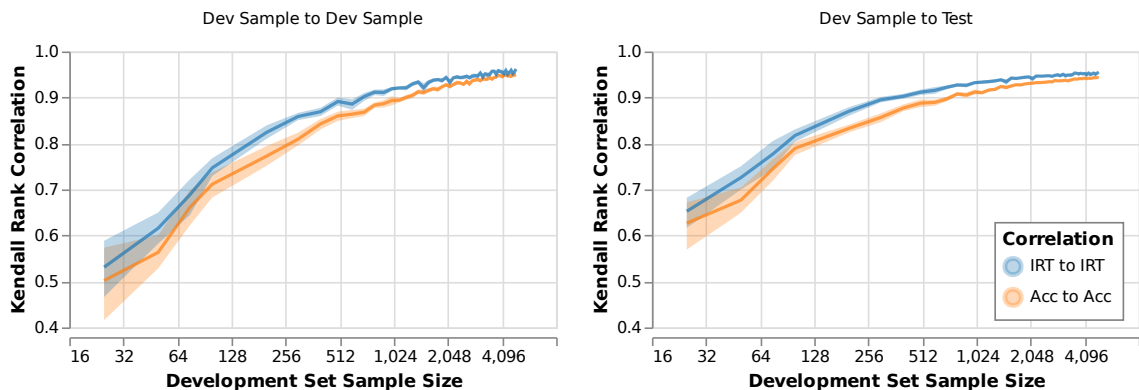


Figure 4.5: Compared to the final ranking over a large test set, how well does a small test set correlate? The left shows correlation between mutually exclusive development set samples and the right between development samples and the full test set. In both experiments (panes), ranking systems by IRT ability is more stable—across all sample sizes—than mean accuracy and thus more reliable (Kendall’s rank correlation is higher). Bands show 95% confidence intervals of rank correlations across ten trials per sample size.

4.4.2.4 What Model Features are Predictive?

Integrating additional features into Bayesian models is not trivial, so we instead use the flexibility of linear models to identify useful features. Our leave-one-in ablation compares features (Figure 4.4): the top ablations both use IRT features, further validating IRT parameters. The subject and item identifier features are also strongly predictive, but item is the stronger of the two. Text-based features are weaker, but this suggests room for future work in better integrating them into IRT models (§4.9).

4.4.3 Ranking with IRT

Leaderboards should produce *reliable* subject rankings: can DAD rank systems even with a tiny test set? Thus, we compare the reliability of traditional average accuracy (§4.3) to IRT ability rankings. Our first experiment (§4.4.3.1) examines the stability of existing items and subjects while the second (§4.4.5) investigates stability of “new” evaluation data using various sampling strategies.

4.4.3.1 IRT Rankings Have Better Reliability

Rankings should be reliable within the same dataset (e.g., on dev set) and generalize to similar datasets (e.g., with a test dataset). To test the first, we measure the ranking stability of mutually exclusive samples of the development data (Buckley and Voorhees, 2000). To test the second, we measure the correlation of between development set sample rankings to test set rankings (Voorhees, 1998).

Specifically, for a range of sample sizes⁸ we (1) sample two partitions of the data, (2) compute the classical ranking⁹ and the IRT ranking from a refit IRT-feas model, then (3) compute Kendall’s correlation (Kendall, 1938) between the samples for each ranking (details in Appendix B.4). In both cases IRT rankings have higher correlation than classical rankings (Figure 4.5, left). Since the benefit is strongest at low sample sizes, IRT can improve the reliability of small-scale evaluations.

The second experiment examines ranking generalization: IRT yields more reliable measures of subject skill, implying a greater consistency in subject rankings across evaluation settings. Figure 4.5 compares the development set sample rankings computed above to rankings obtained using subjects’ *test* set responses (with the same IRT model).

Across all sample sizes, subjects’ IRT ability estimated with the development set has significant correlation with their test set ability. Crucially, this is better than the corresponding classical metrics like accuracy, supporting our original motivation for using IRT.¹⁰

4.4.4 Statistical Significance of Difference in Kendall Tau Coefficients

While Figure 4.5 shows a consistent difference in correlation between ranking methods, it is unclear whether this difference is statistically significant. We estimate the statistical significance of the difference with bootstrap sampling (Efron, 1994).

Since the null case is no difference in correlation coefficients, we seek a symmetric sampling distribution centered at zero that represents a realistic density function. Each ranking stability experiment¹¹ trial results in two lists of number pairs. The lists correspond to subject scores on two datasets;¹² each number pair is the subject’s accuracy and IRT score. To create the bootstrap distribution, we (1) sample with replacement pairs from one list, (2) compute the correlation between the resampled ranking and unused ranking when using accuracy versus IRT score, and (3) compute and store the IRT correlation score minus the accuracy correlation score. We repeat this process 1000 times for each of the 10 trials in the original experiment and aggregate all the differences to build the bootstrap distribution. For each sample size we compute the empirical P-Value on each trial which we show in box and whisker plots (Figure 4.6). While individual P-values tend to be high (particularly for dev to dev sampling), there is still a consistent trend across sample sizes.

⁸The sample size must be less than half the size of the development data so that we can obtain two samples.

⁹For SQUAD, ordering by mean exact match score.

¹⁰Since the maximum trial size was limited, we train one final model with the full data, see Table B.3 in Appendix B.4.

¹¹One experiment for development sample to development sample and one for development sample to test set.

¹²In the first experiment, development set samples; in the second, a development set sample and the full test set.

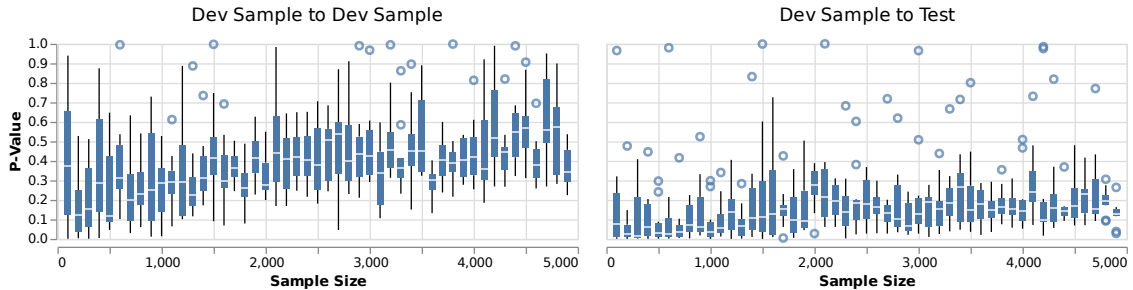


Figure 4.6: P-values of the rank correlation difference for each sample size and trial in Figure 4.5. The inherent noise in dev set sampling makes inferring significance difficult (left); test set driven results (right) are more significant.

4.4.5 IRT Improves Cold Start Reliability

IRT can also guide the construction of tests. Just as IRT practitioners prepare tests for humans, we too construct tests for machines. In educational testing, collecting responses from humans is expensive; likewise, although *questions* are cheap in search-based QA tasks (Nguyen et al., 2016; Kwiatkowski et al., 2019), annotating *answers* is expensive. Likewise, “grading” machine dialog responses is expensive and IRT helps (Sedoc and Ungar, 2020). To emulate this setting, we use computerized adaptive testing (Weiss and Kingsbury, 1984) to iteratively select SQuAD items to “annotate.”

As in human test preparation, we use existing annotations to infer item parameters and iteratively infer the ability of new subjects. This experiment splits m subjects into a training group (80%) and a testing group (20%). The training group represents subjects for which we have full item predictions and annotations; the testing group represents a new group of subjects that we need to rank. To efficiently rank, we should iteratively choose items to annotate that yield the most information about the ranking if all the data were annotated.

This experiment compares how well several item selection strategies work. For each selection method, we (1) choose a sample size, (2), sample from the development set, (3) compute the ranking of subjects, and (4) compute Kendall’s rank correlation (Figure 4.7).¹³

Which item selection strategies should we compare? As a baseline, we use naïve random sampling. Like prior work, we compare selecting items with the highest difficulty and the highest discriminability (Lalor et al., 2019) as well as the sum of the two.¹⁴ We propose that items should be selected according to their Fisher information content (Weiss, 1982)

$$I_i(\theta_j) = \frac{(p'_{ij})^2}{p_{ij}(1 - p_{ij})} = \gamma_i^2 p_{ij}(1 - p_{ij}) \quad (4.2)$$

¹³We compute correlations with the complete development set on ten trials to build 95% confidence intervals.

¹⁴We train an IRT-disc model to simplify sampling (e.g., avoiding a tradeoff between feasibility and discriminability).

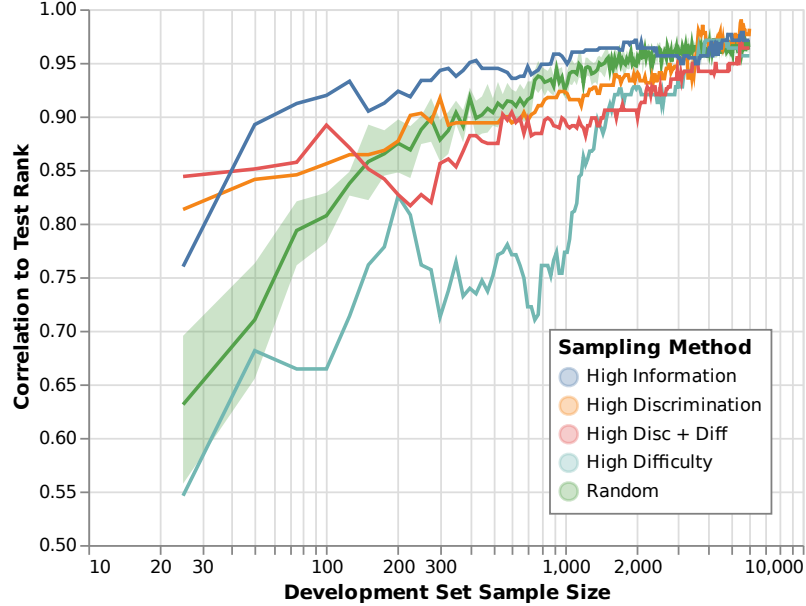


Figure 4.7: Suppose we need to cold start, collect annotations for a new subject: what order would most rapidly increase correlation to the test data? As we expect, the correlations eventually converge, but with little data, IRT has better correlation than other methods. We suspect that the IRT information underperforms early on when the subject ability estimate is unstable.

as derived by Lord et al. (1968, p. 70).

Intuitively, if we do not yet know the true skill θ_j , we should pick items whose expected response we are most uncertain about. Our uncertainty (entropy) is maximized when the likelihood of a correct response p_{ij} is the same as the likelihood of an incorrect response $1 - p_{ij}$, which corresponds to the maximal value of $I_i(\theta_j)$; it is also sensible this value increases as discriminability γ_i increases.

To infer the maximally informative items, we estimate the ability θ_j of each subject using the currently selected items, use the ability to compute the information of each yet-to-be-annotated item for each subject, and then aggregate the informativeness

$$\text{Info}(i) = \sum_j I_i(\theta_j) \quad (4.3)$$

by item i summed over subjects j . This approach is similar to uncertainty sampling and reduces to it for the IRT-base model (Lewis and Gale, 1994). We initially seed with the twenty-five most discriminative items (details in Appendix B.4).

Like computerized adaptive testing (Moreno et al., 1984), Figure 4.7 shows that at lower sample sizes three of the IRT sampling methods are better than random sampling—difficulty does worse. The other IRT methods have comparable correlation. Thus, by using IRT, DAD can both improve rankings and guide annotation.

Discriminability: 3.24 **Difficulty:** 3.86 **Feasibility:** 0 **Mean Exact Match:** 0
Wikipedia Page: [Computational Complexity Theory](#)
Question: In the determination of complexity classes, what are two examples of types of Turing machines?
Official Answer: **probabilistic Turing machines, non-deterministic Turing machines**
Context: Many types of Turing machines are used to define complexity classes, such as deterministic Turing machines, probabilistic Turing machines, non-deterministic Turing machines, quantum Turing machines, symmetric Turing machines and alternating Turing machines. They are all equally powerful in principle, but when resources (such as time or space) are bounded, some of these may be more powerful than others.

Figure 4.8: This question is regarded as infeasible by the IRT model. Upon further inspection, the answer omits five acceptable answers, but more importantly does not permit all combinations of Turing machines.

4.5 Qualitative Insights on Leaderboards

DAD also helps qualitative analysis of items and subjects. First, IRT identifies overfitting and generalizes partitioning datasets by difficulty. Then we show that—like in educational testing—IRT identifies good and bad items.

4.5.1 Guiding Analysis with IRT

Several works curate easy and hard QA subsets based how many models answer correctly (Rondeau and Hazen, 2018) or heuristics (Sugawara et al., 2018). IRT can create similar subsets using IRT-feas, the best 1D model. Difficulty finds where subjects improve while discriminability and feasibility can surface items that may be invalid. For example, one low feasibility question (Figure 4.8¹⁵) asks “what are two examples of types of Turing machines?” which has two problems: (1) the answer omits five types and (2) span-based evaluation precludes selecting non-contiguous types.

After excluding items with negative discriminability—they are likely erroneous—we sort items into bins. We break both difficulty and discriminability into four bins—taking the 25th, 50th, and 75th percentiles—creating eight total bins. Then we select representative SQuAD subjects with their exact match (Figure 4.9). Let’s examine a feasible item with positive difficulty and discriminability like “what reform was attempted following the Nice treaty?”¹⁶ In this case, the annotator’s span is too long—resulting in almost no correct answers and a low fuzzy match score.¹⁷ In contrast, one highly discriminative question succeeds because there are multiple plausible guesses to “who did the Normans team up with in Anatolia?”¹⁸ While

¹⁵Question ID 56e1b00ce3433e14004230a1

¹⁶A: “there was an attempt to reform the constitutional law of the EU and make it more transparent.” (Appendix Figure B.2)

¹⁷For simplicity, we use SQuAD’s dichotomous exact match; SQuAD also has F1—a token-level fuzzy match metric.

¹⁸Example with statistics in Appendix Figure B.3.

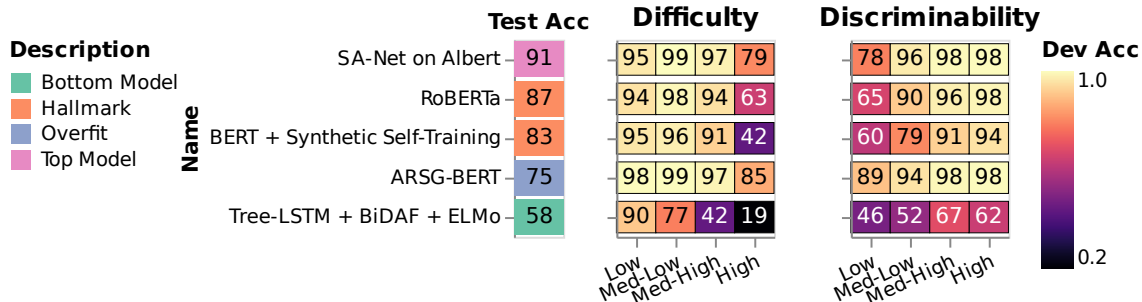


Figure 4.9: We partition evaluation data by IRT difficulty and discriminability with accuracy in each quartile. Most improvements in high-accuracy systems come from getting high-difficulty questions right. Items with low discriminability (and thus prone to annotation errors) are difficult for all subjects except the overfit ARSG-BERT model. We include top-performing SQuAD subjects, several notable subjects (systems), and a pair from the bottom of the leaderboard.

both **the Armenian state** and **Turkish forces** are superficially plausible answers, only **Turkish forces** is correct; nonetheless, some models are fooled. Using IRT to guide subject analysis is helpful; next, we test how efficient it is in identifying annotation error.

4.5.2 Identifying Annotation Error

To test if IRT can identify annotation error we inspect sixty SQuAD development set items. We select ten items from each of these groups: the most negative discriminability, discriminability nearest to zero, the highest discriminability, the least difficult, most difficult, and IRT model errors. For each, we annotate whether the item was correct, was “correct” yet flawed in some way, or simply wrong (Figure 4.10). Inter-annotator agreement between three authors on this three-way annotation with Krippendorff’s α (Krippendorff, 2004; Artstein and Poesio, 2008) is 0.344. Despite only modest agreement, just as in the development of education tests, negative discriminability is predictive of bad items. When discriminability is negative, then the probability of getting the answer right is higher when ability is lower, which is undesirable: Ken consistently loses to Burt on those items. This could identify bad items in evaluation sets for removal.

4.6 Related Work

This chapter draws together two primary threads: we use IRT to understand datasets, which has been applied to other NLP tasks, and apply it to improving leaderboards. Finally, we explore how the insights of IRT can improve not just the analysis of test sets but to improve the *construction* of test sets.

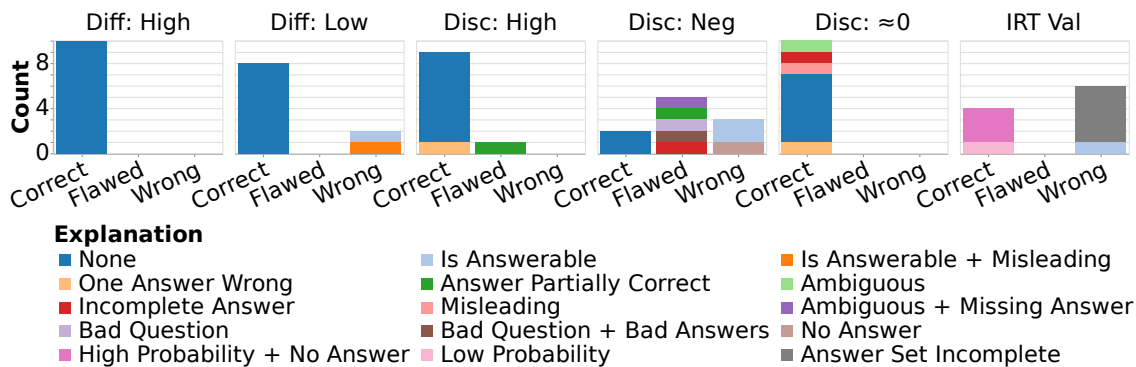


Figure 4.10: We annotate SQuAD items by discriminability, difficulty, and IRT prediction errors. For example, one question with negative discriminability was classified as “Wrong” with the explanation that the annotated answer indicates it is *not* answerable, but the question actually *is* answerable. Items with negative discriminability or where IRT’s prediction is wrong have a much higher rate of annotation error (“Flawed” or “Wrong”). Using similar methodology, errors in datasets could be more rapidly identified.

IRT in NLP IRT is gaining traction in machine learning research (Martínez-Plumed et al., 2016, 2019) where automated metrics can be misleading (Sedoc et al., 2019): machine translation (Hopkins and May, 2013) and chatbot evaluation (Sedoc and Ungar, 2020). Concurrent with our work, Vania et al. (2021) compare NLP test sets with IRT. Closest to our work in NLP is Otani et al. (2016), who rank machine translation subjects and compute correlations with gold scores. Similarly, Martínez-Plumed and Hernández-Orallo (2020) use IRT on non-language AI video game benchmarks. Just as we use IRT to identify difficult or easy items, Lalor et al. (2016) create challenge sets for textual entailment. We test IRT as a way to guide annotation, but it can also be used to train NLP models; for example, deep models learn “easy” examples faster (Lalor et al., 2018) and maintain test accuracy when training data is down-sampled with IRT (Lalor et al., 2019).

Improving Leaderboards The rise of leaderboards in NLP has encouraged critical thought into improving them (Linzen, 2020), improving evaluation more broadly (Eger et al., 2020), and thoughtful consideration of their influence on the direction of research (Sculley et al., 2018; Dotan and Milli, 2020). DAD aims to make leaderboard yardsticks (Hernandez-Orallo, 2020) more reliable, interpretable, and part of curating the benchmark itself. In line with our reliability goal, just as statistical tests should appear in publications (Dror et al., 2018; Dodge et al., 2019), they should be “freebies” for leaderboard participants (Ethayarajh and Jurafsky, 2020). Alternatively, Hou et al. (2019) posit that leaderboards be automatically extracted from publications. Aggregating multi-task (Wang et al., 2019b,a; Fisch et al., 2019) and multi-metric benchmarks (Ma et al., 2021) is an open question which—although we do not address—is one use for IRT.

This chapter implicitly argues that leaderboards should be continually up-

dated. As a (static) leaderboard ages, the task(s) become overfit (Recht et al., 2019) which—although mitigable (Blum and Hardt, 2015; Anderson-Cook et al., 2019)—is best solved by continually collecting new data (Kiela et al., 2021). Ideally, these new data should challenge models through adversarial collection (Wallace et al., 2019b; Nie et al., 2020) and other methods (Gardner et al., 2020a). However, if making a too easy leaderboard harder is not possible, we should recognize that the leaderboard has outlived its helpfulness and should be retired (Voorhees, 2000b).

Part of this chapter centers on alternate task efficacy rankings, but this naïvely assumes that task efficacy is the sole use case of leaderboards. Indeed, focusing solely these factors can mislead the public (Paullada et al., 2020), may not reflect human language capabilities (Schlangen, 2020), or even be ecologically valid (de Vries et al., 2020). Leaderboards are also well positioned to provide incentive structures for participants that prioritize fairness (Bender and Friedman, 2018) and efficiency (Strubell et al., 2019; Schwartz et al., 2020; Min et al., 2021) or incorporate testing of specific capabilities (Ribeiro et al., 2020; Dunietz et al., 2020). To enable these more nuanced analyses, leaderboards should accept runnable models rather than static predictions (Ma et al., 2021).

Active Learning Beyond IRT, the analysis of training dynamics and active learning (Settles, 2009) is helpful for actively sampling specific items or identifying low-quality items (Brodley and Friedl, 1999). For example, Swayamdipta et al. (2020) and Pleiss et al. (2020) propose alternative training dynamics-based methods for identifying difficult items as well annotation errors. Even closer to goals, Rahman et al. (2020) use active learning to build a test collection. Explicitly measuring how effectively examples separate the best subject from the rest allows test set curators to “focus on the bubble” (Boyd-Graber and Börschinger, 2020), prioritizing examples most likely to reveal interesting distinctions between submitted systems.

Alternate Formulations IRT is an example of convergent evolution of models that predict subject action given an item. Ideal point models (Poole and Rosenthal, 2017) consider how a legislator (subject) will vote on a bill (item) and use a similar mathematical formulation. The venerable ELO model (Glickman and Jones, 1999) and modern extensions (Herbrich et al., 2007) predict whether a player (subject) will defeat an opponent (item) with, again, a similar mathematical model. Certain IRT models can also be formulated as nonlinear mixed models (Rijmen et al., 2003), where the item parameters are fixed effects and the latent subject parameters are random effects. This allows for comparisons between IRT models and other mixed-effects models under a consistent framework. **IRT-base** and **IRT-disc** can be formulated as nonlinear mixed models, and **IRT-feas** can be formulated as a discrete mixture model over items. As we discuss further in the next section, DAD’s application of IRT can be further improved by adopting interpretable extensions of these models.

4.7 Conclusion

This chapter advocates incorporating decades of research in crafting education tests to improve how we evaluate the capabilities of NLP models. We propose and validate an alternate IRT ranking method for leaderboard evaluations, show it can guide annotation, detect annotation error, and naturally partition evaluation data. Just as educators moved from classical testing to IRT, the NLP community should consider future evaluations with IRT.

4.8 Limitations

Although there is much to gain through IRT evaluation, there are limitations which make it hard to implement. First, it requires access to item-level responses for all examples for all subjects which are often only available to organizers. Second, [Urbano \(2016\)](#) notes that sampling mutually exclusive subsets has drawbacks—samples are not entirely independent. Lastly, our work is a proof of concept using SQuAD 2.0 as a test bed, and our results may not generalize.

4.9 Future Work

We see a few directions for future work which we outline here and describe further in Chapter 7. First, this work validates IRT’s usefulness for leaderboards, but to fully realize its potential IRT needs to be implemented in an existing or new benchmark. Second, our IRT models do not incorporate the item content (e.g., question text) when predicting responses, but in principle could; Bayesian models with metadata ([Card et al., 2018](#)) and ideal point models from political science ([Poole and Rosenthal, 1985](#)) that incorporate bills and speeches do exactly this ([Gerrish and Blei, 2011](#); [Nguyen et al., 2015](#); [Kraft et al., 2016](#)). Analogously, IRT for leaderboards can and should also incorporate text from passages, questions, and answers to better model what makes questions difficult. Such a model can also predict which characteristics would create discriminating or difficult items. Third, statistical testing is strongly recommended in NLP ([Dror et al., 2018](#); [Dodge et al., 2019](#)), and we should investigate how IRT based statistical tests differ (§B.5). Lastly, multidimensional IRT models to evaluate multiple skills could aid multi-task leaderboards like MRQA ([Fisch et al., 2019](#)) and Dynaboard ([Ma et al., 2021](#)).

While this chapter leaves the dataset and format unchanged, the broader message is that we should understand how question difficulty and discriminability affect QA evaluations. In the next two chapters, we introduce a Manchester paradigm QA format that—by construction—has more discriminative power (Chapter 5) and use human-machine cooperative writing to create more challenging questions (Chapter 6).

Chapter 5: The Case for Incremental Question Answering Evaluation

You'll rarely hear it acknowledged on *Jeopardy!* or *Who Wants to Be a Millionaire*, but many of the elite winners on these shows got their trivia training in high school and college quiz bowl. . . When I took the written test at my *Jeopardy!* audition. . . I felt like a runner who'd been training in high altitude Mexico City, just to get his lungs in such superpowered shape that events at lower elevations seemed like a piece of cake.

Brainiac: Adventures in the Curious, Competitive, Compulsive World of Trivia Buffs
Ken Jennings (2006, p. 81)

In contrast with asking questions to learn (Chapter 3), scholastic trivia competitions use questions to test knowledge and intelligence.¹ In this way and in line with the Manchester paradigm, trivia games are a variant of the Turing test since we expect that answering a set of questions as well as a human is a minimum bar towards demonstrating human-like intelligence (Yampolskiy, 2013). In the previous chapter, we considered a task (SQuAD) where participants are scored dichotomously:² they are either correct or wrong. A byproduct of this choice is that when subjects are both correct or both wrong, we learn substantially less about their latent skill (§4.4.5). Unfortunately, as Ray Hamel³ notes “the problem of gauging difficulty, of making a question easy enough to be accessible, but tough enough so that listeners still have to scratch their heads” (Jennings, 2006, p. 228) is challenging.

This chapter makes the case that the format used by the trivia game Quizbowl (QB) better distinguishes between players (machine or human) of disparate skill while incorporating dynamic elements that also make it fertile space for sequential decision-making in QA.⁴ Beyond the format of QB, we demonstrate how collabora-

¹This chapter is based on Rodriguez et al. (2019).

²SQuAD’s fuzzy matching metrics are arguably a concession to annotation noise, rather than genuine partial credit.

³Ray Hamel is a member of the Trivia Bowl Hall of Fame, estimates he has written over 40,000 trivia questions, and has written over 750 (trivia) crossword puzzles for Games, The New York Times, The Los Angeles Times, Newsday, Dell Champion Crosswords, and CrosSynergy (Hamel, 2000).

⁴This work substantially expands the number questions compared to Boyd-Graber et al. (2012) and player-question records compared to He et al. (2016b). We also make the setting significantly harder by not restricting questions to only the most frequently asked about answer (1K versus 24K) as in Iyyer et al. (2015). This work also incorporates the exhibition match from Boyd-Graber et al.

At its premiere, the librettist of this opera portrayed a character who asks for a glass of wine with his dying wish. That character in this opera is instructed to ring some bells to summon his love. At its beginning, a man who claims to have killed a serpent has a padlock put on his mouth because of his lying. The plot of this opera concerns a series of tests that Tamino must undergo to rescue Tamina from Sorastro. For 10 points, name this Wolfgang Mozart opera titled for an enchanted woodwind instrument.
Answer: The Magic Flute

Figure 5.1: QB is a trivia game where questions begin with clues that are initially difficult, but become progressively easier until a giveaway at the end of the question. Players answer as soon as they know the answer so as a result the earlier they answer the more knowledgeable they are. For example, answering after the first sentence indicates the player recognizes the librettist (Emanuel Schikaneder) and knows that they played Papageno in *The Magic Flute* (die Zauberflöte). Answering at the end of the question only requires surface knowledge of Mozart’s opera works.

tion with a thriving and enthusiastic community provides natural ways to engage with and educate the public about NLP research; Chapter 6 takes this a step farther by combining the efforts of humans and machines to author better QA datasets.

5.1 An Introduction to Quizbowl

Answering questions is an important skill for both humans and computers. Exams form the foundation of educational systems and—for many societies—of the civil system (Fukuyama, 1995). Computers answering questions in the Turing test is the standard definition of artificial intelligence (Turing, 1950). But another more trivial form of question answering is more pervasive in popular culture.

Trivia games are pervasive and popular: from quizzing in India (Roy, 2016) to “What? Where? When?” in Russia (Korin, 2002) to “Who wants to be a Millionaire” (Clarke et al., 2001; Lam et al., 2003), trivia encourages people to acquire, recall, and reason over facts. For computers, Yampolskiy (2013) argues that these skills are AI-complete: solve question answering and you have solved AI generally. A central idea in this chapter is that the intense research in question answering would benefit in adopting the innovations and lessons learned from human trivia competition, as embodied in a trivia format called **Quizbowl** (QB).

In QB, questions are posed *incrementally*—word by word—and players must *interrupt* the question when they know the answer (Figure 5.1). “Speed is therefore crucial” since it rewards players who can answer with less information than their opponents (Jennings, 2006). This is not just a gimmick to separate it from other question answering formats: players must simultaneously think about what is the

(2017). Finally, we create a new evaluation procedure that better estimates how models fare in the real-world versus human opponents.

most likely answer and after every word decide whether it is better to answer or wait for more information. To succeed, players and machines alike must answer questions, maintain accurate estimates of their confidence, and factor their opponents’ abilities. The combination of these skills makes QB challenging for machine learning algorithms.

A dedicated and skilled community forged QB over decades (§5.2), creating a diverse and large dataset (§5.3). We refer to this dataset as the QANTA dataset because (in our opinion) **Q**uestion **A**nswering is **N**ot a **T**rivial **A**ctivity.⁵

Playing QB requires deciding *what* to answer (§5.5) and *when* to answer (§5.6). Our final contribution is a framework that combines independent systems for each of these sub-tasks. Despite its simplicity, our implementation of this framework is competitive with the best players. Section 5.8 showcases QB as a platform for simultaneously advancing research and educating the public about the limits of machine learning through live human–computer competitions. Finally, we discuss ongoing and future research using trivia questions to build machines that are as capable of reasoning and connecting facts as humans.

5.2 Why Quizbowl?

When discussing machine learning and trivia, the elephant in the room is always IBM’s tour-de-force match (Ferrucci et al., 2010) against Ken Jennings and Brad Rutter on *Jeopardy!* Rather than ignore the obvious comparisons, we take this on directly and use the well-known *Jeopardy!* context—which we gratefully acknowledge as making our own work possible—as a point of comparison, as QB is a better differentiator of skill between participants, be they human or machine (§5.2.1 and §5.2.2). Throughout, we discuss the history of trivia to show that the hard-won lessons humans learned about question answering transfer to machine question answering.

The QA format categorization of Gardner et al. (2020b) names three tasks where framing the problem as QA is *useful*: (1) filling human information needs, (2) QA as annotation or probe, and (3) as a transfer mechanism.⁶ Like SearchQA (Dunn et al., 2017, Jeopardy!), QB does not explicitly probe specific linguistic phenomena; it uses language to ask what humans know. In contrast to questions posed to search engines or digital assistants (Nguyen et al., 2016; Kwiatkowski et al., 2019), QB is less ambiguous: question writers ensure that the descriptions uniquely identify one and only one answer, a non-trivial goal (Voorhees and Tice, 2000). Thus, although the goals in QB are superficially similar to open domain information-seeking, they are distinct and align with the Manchester paradigm (§2.1.2).

The QB format is compelling and consistent because of its evolution (Figure 5.2) over its fifty-year history.⁷ Many of the challenges the NLP community

⁵Dataset available at <http://datasets.qanta.org>.

⁶The Cranfield and Manchester paradigms (§2.1.1 and §2.1.2) are related to (1) and (2).

⁷After returning from World War II and inspired by USO morale-building activities, Canadian Don Reid sketched out the format with the first host Allen Ludden. After a radio premiere

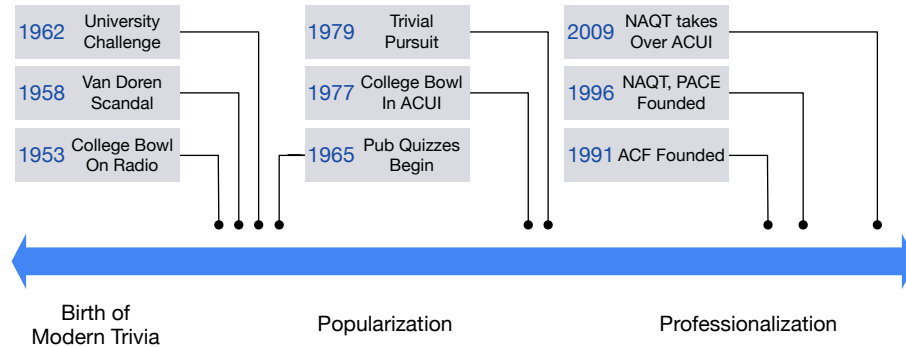


Figure 5.2: Trivia has gone from a laid-back pastime to an organized, semi-professional competition format. The QB framework, in particular, which arose from College Bowl (US) and University Challenge (UK) emphasizes fairness and the ability to discover the better question answerer. As organizations such as the Academic Competition Federation and National Academic Quiz Tournaments emerged, the format has focused on academic, well-run tournaments.

faces in collecting good question answering datasets at scale (Hermann et al., 2015) were first encountered by trivia aficionados. For example, avoiding predictive yet useless patterns in data (Jia and Liang, 2017; Kaushik et al., 2020); players do not like re-used clues making questions trivially easy. Similarly, players prize the novelty of questions which is often created by

tak[ing] two elements you wouldn’t have thought had a commonality and put[ing] them together and then you go, ‘Oh, yea, that’s pretty cool, there’s a connection there that nobody’s seen.’ (Ray Hamel (Jennings, 2006))

Creating multi-hop reasoning questions is a shared goal with HotPotQA (Yang et al., 2018b), but it has been challenging to write questions that actually require it (Min et al., 2019). We distill these lessons, describe the craft of question writing that makes QB a compelling question answering task (§5.2.3), and enumerate some NLP challenges required to truly solve QB (§5.2.4). We conclude by framing QB as a hybrid task between question answering and sequential decision-making (§5.2.5).

5.2.1 What is a Buzzer Race?

The scapegoat for every *Jeopardy!* loser and the foundation of every *Jeopardy!* winner is the **buzzer** (Harris, 2006). A buzzer is a small handheld device that players

in 1953, *College Bowl* moved to television in 1959 and became the first television show to win a Peabody (Baber, 2015). The format established many careers: the future president of the National Broadcasting Corporation (NBC), Grant Tinker, served as the game’s first scorekeeper (the newly designed game and its scoring was so confusing that Allen Ludden often had to *ad lib* to let Tinker catch up). The format was intriguing enough that Granada studios copied it—initially without permission—into what became the UK cultural touchstone *University Challenge* (Taylor et al., 2012), establishing the career of Bamber Gascoigne.

press to signal that they can correctly respond to a clue. The fundamental difference between *Jeopardy!* and QB—and what makes QB more suitable for research—is how clues are revealed and how players use the buzzer.

Jeopardy! is a television show and uses the buzzer to introduce uncertainty, randomness, and thus excitement for the viewer at home.⁸ In *Jeopardy!*, players can only use the buzzer when the moderator has finished reading the question.⁹ If players use the buzzer before the question is finished, they are locked out and prevented from answering the question for a fraction of a second (an eternity in the fast-paced game of *Jeopardy!*).

This advantaged Watson in its match against two opponents with feeble human thumbs and reflexes, as *Jeopardy!* uses the buzzer to determine who among those who know the answer *has the fastest reflexes*.¹⁰ While Watson gets an electronic signal when it was allowed to buzz, the two humans watch for a light next to the *Jeopardy!* game board to know when to buzz. Thus, Watson—an electronic buzzing machine—snags the first choice of questions, while the two humans fight over the scraps. In *Jeopardy!* reflexes are almost as important as knowledge. Next we show how the structure of QB questions and its use of a buzzer rewards depth of knowledge rather than reflexes.

5.2.2 Pyramidality and Buzzers

In contrast, QB is a game honed by trivia enthusiasts which uses buzzers as a tool to determine *who knows the most about a subject*. This is possible because the questions are *interruptable*. Unlike *Jeopardy!*, players can interrupt the questions when they know the answer (recall questions are multi-sentence in QB). This would make for bad television (people like to play along at home and cannot when they cannot hear the whole question), but makes for a better trivia game that also requires decision-making under uncertainty.

This alone is insufficient however; if an easy clue appears early in the question then knowing hard clues later in the question is irrelevant. Questions that can be answered with only a fraction of their input are a bad foundation for research (Sugawara et al., 2018; Feng et al., 2019). QB addresses this problem by structuring questions *pyramidally*. In pyramidal questions, clues are incorporated so that harder, more obscure information comes first in the question, and easier, more obvious information comes at the end of the question. Thus, when a player answers before their opponents, they are more knowledgeable than their opponents.

This also makes QB an attractive machine learning research domain. The

⁸As Jennings (2006) notes about QB, “certainly it’s hard to play along at home with a game where the questions can be interrupted after the moderator has only read a few syllables.”

⁹In *Jeopardy!* terminology is reversed so that a moderator reads clues termed *answers* to which players must supply the correct *question*. To avoid confusion, we follow standard terminology.

¹⁰In a Ken Jennings interview with NPR (Malone, 2019), the host Kenny Malone summarized it well as “To some degree, *Jeopardy!* is kind of a video game, and a crappy video game where it’s, like, light goes on, press button—that’s it.” Ken Jennings agreed, but characterized it as “beautiful art and not a really crappy video game”.

giveaways are often easy for computers too: they are prominent on Wikipedia pages and have appeared in many questions. Thus, it is easy for computers to answer most questions *at some point*: QB is not an impossibly difficult problem. The challenge then becomes to answer the questions *earlier*, using more obscure information and higher-order reasoning.

Humans who play QB have the same yearning; they can answer most of the questions, but they want to deepen their knowledge to buzz in just a little earlier. They keep practicing, playing questions and going to tournaments to slowly build skill and knowledge. QB is engineered for this to be a rewarding experience.

The same striving can motivate researchers: it does not take much to buzz in a word earlier. As small incremental improvements accumulate, we can have more robust, comprehensive question answering systems. And because QB has a consistent evaluation framework, it is easy to see whose hard work has paid off.

Thus, the form of QB questions—the product of decades of refining how to measure the processing and retrieval of information of humans—can also compare machines’ question answering ability. We next describe the cultural norms of question writing in the QB community that contribute to making it a challenging task for humans and machines alike.

5.2.3 The Craft of Question Writing

The goal of QB is to reward “real” knowledge. This goal is the product of a long history that has resulted in community norms that have evolved the competition into a thriving, carefully designed trivia ecosystem. By adopting these conventions, machine learning can benefit from the best practices for question answering evaluation without repeating the same mistakes.

Every year, question writers in the community focus on creating high quality questions that are novel and pyramidal. Experts write thousands of questions each year.¹¹ To maintain the quality and integrity of competition, the community enforces rules consistent with machine learning’s question for generalization as described by [Boyd-Graber and Börschinger \(2020\)](#): avoiding ambiguity, ensuring correctness, eschewing previously used clues, and allowing for fair comparisons between teams ([Lujan and Teitler, 2003](#); [Vinokurov](#); [Maddipoti](#)) of 10,000 middle school students, 40,000 high school students, and 3,200 college students ([National Academic Quiz Tournaments, 2020](#)). At the same time, in preparation for tournaments students study questions from previous years.

These dueling groups—players and writers—create a factual arms race that is the foundation for the quality of QB questions. Aligning annotators’ motivations ([von Ahn, 2006](#))—such as playing a game—with the goals of the data collection improves the quality and quantity of data. A similar arms race between dataset

¹¹Regional competition questions are written by participants; championship competition questions are written by professionals hired by either the Academic Competition Federation (ACF), National Academic Quiz Tournaments (NAQT), or the Partnership for Academic Competition Excellence (PACE). While the exact organizational structure varies, initial draft questions are vetted and edited by domain experts.

exploiters (attackers) and those seeking to make datasets more robust (defenders) exists in other machine learning domains like computer vision (Carlini and Wagner, 2017; Hendrycks et al., 2021) and Build-It, Break-It, (Fix-It) style tasks (Ettinger et al., 2017; Thorne et al., 2019; Dinan et al., 2019a; Nie et al., 2020).

In QB, answers are uniquely identifiable named entities such as—but not limited to—people, places, events, and literary works. These answers are “typified by a noun phrase” as in Kupiec (1993) and later in the TREC QA track (Voorhees, 2003a). Similar answer types are also used by other factoid question answering datasets such as SimpleQuestions (Bordes et al., 2015), SearchQA (Dunn et al., 2017), TriviaQA (Joshi et al., 2017), and NaturalQuestions’ short answers (Kwiatkowski et al., 2019). In its full generality, QB is an Open Domain QA task (Chen et al., 2017; Chen and Yih, 2020). However, since the vast majority of answers correspond to one of the six million entities in Wikipedia (§5.3.4),¹² we approximate the open-domain setting by defining this as our source of answers (Section 5.9.1 reframes this in reading comprehension’s span selection format). Like the ontology of ImageNet (Deng et al., 2009), no formalism is perfect, but it enables automatic answer evaluation and linking to a knowledge base. In QB though, the challenge is not in framing an answer, it is in answering at the earliest possible moment.

The pyramidal construction of questions—combined with incrementality—makes QB a more fair and granular comparison. For example, the first sentence of Figure 5.1—also known as the lead in—while obscure, uniquely identifies a single opera. Questions that begin misleadingly are scorned and derided in online discussions as “neg bait”;¹³ Thus, writers ensure that all clues are uniquely identifying even at the start.

The entirety of questions are carefully crafted, not just the lead-in. Middle clues reward knowledge but cannot be too easy: frequent clues in questions or clues prominent in the subject’s Wikipedia page are considered “stock” and should be reserved for the end. These same insights have been embraced by machine learning in the guise of adversarial methods (Jia and Liang, 2017) to eschewing superficial pattern matching. In contrast, the final giveaway clue should be direct and well-known enough; someone with even a passing knowledge of The Magic Flute would be able to answer.

This is the product of a complicated and nuanced social dynamic in the QB community. Top teams and novice teams often play on the same questions; questions are—in part—meant to teach (Gall, 1970) so are best when they are fun and fair for all. The pyramidal structure ensures that top teams use their deep knowledge and quick thinking to buzz on the very first clues, but novice teams are entertained

¹²A minority of answers cannot be mapped. Some answers do not have a page because Wikipedia is incomplete (e.g., not all book characters have Wikipedia pages). Other entities are excluded by Wikipedia editorial decisions: they lack notability, are combined with other entities (e.g., Gargantua and Pantagruel and Romulus and Remus). Other abstract answers will likely never have Wikipedia pages (women with one leg, ways Sean Bean has died in films).

¹³“Negging” refers to interrupting a question with a wrong answer; while wrong answers do happen, a response with a valid chain of reasoning should be accepted. Only poorly written questions admit multiple viable answers.

and learning until they get to an accessible clue. Just about everyone answers all questions (it is considered a failure of the question writer if the question “goes dead” without an answer).

QB is not just used to test knowledge; it also helps discover new information and as a result diversifies questions (“oh, I did not know the connection between the band the Monkees and correction fluid!”).¹⁴ While most players will not recognize the first clue (otherwise the question would not be pyramidal), it should be interesting and connect to things the player would care about. For example, in our Magic Flute question, we learn that the librettist appeared in the premiere, a neat bit of trivia that we can tuck away once we learn the answer.

These norms have established QB questions as a framework to both test and educate human players. Our thesis is that these same properties can also train and evaluate machine question answering systems. Next, we highlight the NLP and ML challenges in QB.

5.2.4 Quizbowl for Natural Language Processing Research

We return to Figure 5.1, which exemplifies NLP challenges common to many QB questions. We already discussed (*pyramidity*): each sentence uniquely identifies the answer but each is easier than the last. The most knowledgeable answers earlier and “wins” the question. But what makes the question difficult apart from obscurity (Boyce-Jacino and DeDeo, 2018)? Answering questions early is significantly easier if machines can resolve coreference (Ng, 2010) and entity linking (Shen et al., 2015).

First, the computer should recognize “the librettist” as Schikaneder, *whose name never appears in the question*. This special case of entity linking to knowledge bases is sometimes called Wikification (Cheng and Roth, 2013; Roth et al., 2014). The computer must recognize that “the librettist” refers to a specific person (mention detection), recognize that it is relevant to the question, and then connect it a knowledge base (entity linking).

In addition to linking to entities *outside* the question, another challenge is connecting coreferences within a question. The interplay between coreference and question answering is well known (Stuckardt, 2003), but Guha et al. (2015) argue that QB coreference is particularly challenging: referring expressions are longer and oblique, world knowledge is needed, and entities are named *after* other referring expressions. Take the character Tamino (Figure 5.1): while he is eventually mentioned by name, it is not until after he has been referred to obliquely (“a man who claims to have killed a serpent”). The character Papageno (portrayed by Schikaneder) is even worse; while referred to twice (“character who asks for a glass of wine”, “That character”), Papageno is never mentioned by name. To fully solve the question, a model may have to solve a difficult coreference problem **and** link the reference to Papageno and Schikaneder.

¹⁴Bette Nesmith Graham, the mother of Monkees band member Michael Nesmith, invented correction fluid in 1956.

These inferences, like in the clue about “the librettist”, are often called *higher-order reasoning* since they require creating and combining inference rules to derive conclusions about multiple pieces of information (Lin and Pantel, 2001). Questions that require only a single lookup in a knowledge base or a single IR query are uninteresting for both humans and computers; thus, they are shunned for QB lead-in clues. Indeed, the first sentences in QB questions are the most difficult clues for humans and computers because they often incorporate surprising, quirky relationships that require skill and reasoning to recognize and disentangle. Interest in multi-hop question answering led to the creation WikiHop through templates (Welbl et al., 2018) and HotPotQA through crowdsourcing (Yang et al., 2018b). In contrast to these artificially or crowdsourced created datasets, QB questions focus on links that experts view as relevant and important.

Finally, even the final clue (called a “giveaway” because it’s so easy for humans) could pose issues for a computer. Connecting “enchanted woodwind instrument” to The Magic Flute requires solving wordplay. While not all questions have all of these features, these features are typical of QB questions and showcase their richness.

Crowdsourced datasets like OpenBooksQA (Mihaylov et al., 2018) and CommonSenseQA (Talmor et al., 2019) have artifacts that algorithms can game (Geva et al., 2019): they find the right answer for silly reasons. For example, answering correctly with just a handful of words from a SQuAD question (Feng et al., 2018), none of a bAbI question (Kaushik and Lipton, 2018), or the image in a question about an image (Goyal et al., 2017). Although the QANTA dataset and other “naturally occurring” data likely do contain machine exploitable patterns, they do not face the same quality issues since the author’s motivation is intrinsic: to write an entertaining and educational question as in QB.

5.2.5 Quizbowl for Machine Learning Research

While answering questions showcases the NLP challenges, deciding *when* to answer showcases the ML challenges related to decision theory (Raiffa, 1968). As in games like Poker (Brown and Sandholm, 2019), QB players have incomplete information: they do not know when their opponent will answer, do not know what clues will be revealed next, or if they will know the next clues. In our buzzer model, the QA model output is but one piece of information used to make the decision—under uncertainty—of when to buzz in. Since a decision must be made at every time step (word), we call this an incremental classification task.

We formalize the incremental classification task as a Markov Decision Process (Zubek and Dietterich, 2002, MDP). The actions in this MDP correspond to what a player can do in a real game: click the buzzer and provide their current best answer or wait (one more word) for more information. The non-terminal states in the state space are parameterized by the text of the question revealed up to the current time step, the player’s current best guess, and which player (if any) has already buzzed incorrectly. Rewards are only given at terminal states and transitions to those states are determined by which player correctly answered first. Additionally, we treat the opponent as a component of the environment as opposed

to another agent in the game.¹⁵ This task—the buzzing task—has connections to work in model confidence calibration offline (Yu et al., 2011; Nguyen and O’Connor, 2015) as well as online (Kuleshov and Ermon, 2017), cost-sensitive learning (Elkan, 2001), acquisition of features with a budget (Lizotte et al., 2003), and incremental classification (Melville et al., 2005).

For humans, effective QB play involves maintaining a correctness estimate of their best answer, weighing the cost and benefits of answering now versus waiting, and making buzzing decisions from this information. Naively, one might assume that model calibration is as simple as examining the probability output by the (neural) QA system, but neural models are often especially poorly calibrated (Guo et al., 2017) and calibrations often fail to generalize to out of domain test data (Kamath et al., 2020). Since QB training data spans many years, models must also contend with domain shift (Ovadia et al., 2019). Model calibration is naturally related to deciding when to buzz—also known as answer triggering in QA and information retrieval (Voorhees, 2001; Yang et al., 2015).

Unlike standard answer triggering though, in QB the expected costs and benefits are continually changing. Specifically, there are costs for obtaining new information (seeing more words) and costs for misclassifications (guessing incorrectly or waiting too long). This parallels the setting where doctors iteratively conduct medical tests until they are confident in a patient’s diagnosis (Zubek and Dietterich, 2002; Chai et al., 2004).

Although this can be framed as reinforcement learning, we instead frame buzzing in Section 5.6 as incremental classification as in Trapeznikov and Saligrama (2013). In this framing, a binary classifier at each time step determines when to stop obtaining new information and render the decision of the underlying (QA) model. As Trapeznikov and Saligrama (2013) note, evaluation in this scenario is conceptually simple: compare the costs incurred to benefits gained.

Evaluation We evaluate the performance of our systems through a combination of standalone comparisons (Section 5.7.1) and simulated QB matches (Section 5.7.3). For standalone evaluation we incrementally feed systems new words and record their responses. We then calculate accuracy for each position in the question (e.g., after the first sentence, halfway through the question, and at the end). While standalone evaluations are useful for developing systems, the best way to compare systems and humans is with evaluations that mimic QB tournaments.

A recurring theme is our mutually beneficial collaboration with the QB community: host outreach exhibitions (Section 5.8), annotate data, play with and against our systems (Section 5.10.2), and collect the QANTA dataset. This community created this rigorous format for question answering over decades and continues to help understand and measure the question answering abilities of machines.

¹⁵This is not precisely true in our live exhibition matches; although we treat the opponent as part of the environment, our human opponents do not and usually adapt to how our system plays. For instance, it initially had difficulty with pop culture questions.

5.3 The QANTA Dataset

This section describes the QANTA dataset from the QB community (§5.3.1). The over 100,000 human-authored, English questions from QB trivia tournaments (§5.3.2) allows systems to learn what to answer. More uniquely, 3.9 million filtered records of humans playing QB online (§5.3.3) allows systems learn when to “buzz in” against opponents (§5.4).

5.3.1 Sources of Quizbowl Questions

The QB community maintains and curates several public databases of questions spanning 1997 to today.¹⁶ On average, 10,000 questions are written every year. Our dataset has 119,247 questions with over 650 thousand sentences and 11.4 million tokens.

To help players practice and to build a dataset showing how humans play, we built the first website for playing QB online (Figure 5.3a). After initial popularity, we shut down the site; however, enterprising members of the QB community resurrected and improved the application. 89,477 players used the successor (Figure 5.3b) and have practiced 5.1 million times on 131,075 unique questions. A filtered¹⁷ subset of 3.9 million player records forms the second component of our dataset, which we call **gameplay data**.

5.3.2 QANTA Questions

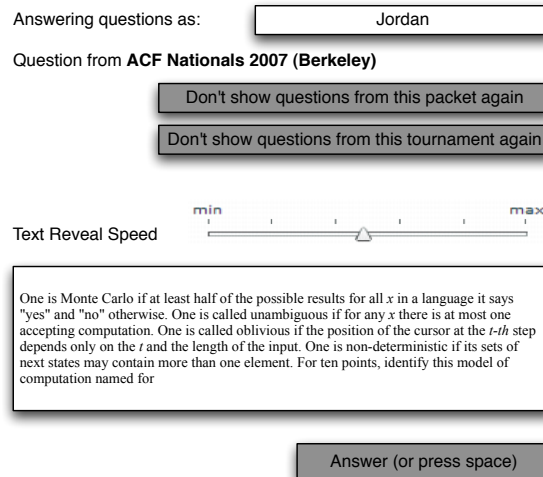
Table 5.1 compares QA datasets written by humans. Because often each QB sentence has enough information for players to answer, each QANTA instance can be broken into four to six pseudo sentence-answer pairs. Although our dataset does not have the most questions, it is significantly larger in the number of sentences and tokens.

In addition to QANTA having more sentences, questions are longer (Figure 5.4), especially compared to crowd-sourced datasets. As a side effect of both being longer and not crowdsourced, QB sentences are syntactically complex and topically diverse (Figure 5.5).

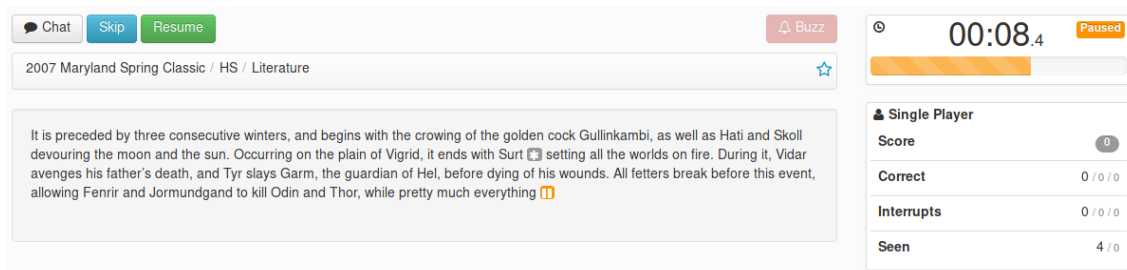
Topical Diversity of Questions Creating diverse datasets is a shared goal between researchers developing NLP resources and organizers of QB tournaments. QB questions are syntactically diverse with dense coreference (Guha et al., 2015) and cover a wide range of topics. Diversity takes the form of questions that reflect the topical, temporal, and geographical breadth of a classical liberal education. For example, the Academic Competition Federation mandates that literature cover American, British, European, and world literature (Vinokurov et al., 2014). Moreover,

¹⁶Questions in were obtained (with permission) from <http://quizdb.org> and <http://protobowl.com>.

¹⁷We include only a player’s first play on a question and exclude players with less than twenty questions.



(a) Our 2012 interface was the first way to play QB online.



(b) The QB interface for collecting most of our gameplay records. It improved over our own through features like real-time competitive play and chatrooms.

Figure 5.3: Our interface and a popular modern interface for playing QB online. Both interfaces reveal questions word-by-word until a player interrupts the system and makes a guess.

authors must “vary questions across time periods”—with no more than one post 1990 literature—and questions must “span a variety of answers such as authors, novels, poems, criticism, essays, etc.” There are similarly detailed proscriptions for the rest of the distribution.

Figure 5.5 shows the category and sub-category distribution over areas such as history, literature, science, and fine arts. Taken together, QB is a topically diverse dataset across broad categories and finer-grained sub-categories. This diversity contrasts with a sample of 150 questions from NaturalQuestions (Kwiatkowski et al., 2019)¹⁸ which indicates that questions are predominantly about Pop Culture (40%), History (19%), and Science (15%); see Appendix C.2 for complete results. This emphasizes that to do well, players and systems need to have both breadth and depth of knowledge.

¹⁸The authors annotated 150 questions from the development set using the same categories as QB.

Dataset	QA Pairs	Tokens
SimpleQuestions (Bordes et al., 2015)	100K	.614M
TriviaQA (Joshi et al., 2017)	95K	1.21M
SQuAD 1.0 (Rajpurkar et al., 2016)	100K	.988M
SearchQA (Jeopardy!) (Dunn et al., 2017)	216K	4.08M
QANTA 2012 (Boyd-Graber et al., 2012)	47,610 / 7,949	1,073,085
QANTA 2014 (Iyyer et al., 2014)	163,667 / 30,658	4,009,059
QANTA 2018 (This Work)	650K / 120K	11.4M

Table 5.1: The QANTA dataset is larger than most question answering datasets in QA pairs (120K). However, for most QB instances each sentence in a question can be considered a QA pair so the true size of the dataset is closer to 650K QA pairs. In Section 5.5 using sentence level QA pairs for training greatly improves model accuracy. The QANTA dataset has more tokens than all other QA datasets. Statistics for QANTA 2012 and 2013 only include publicly available data.

Answer Entity Type Diversity QB questions are also diverse in the kinds of entities that appear as answers (25K entities in the training data). A dataset which is topically diverse, but only asks about people is not ideal. Using the Wikidata knowledge graph we obtain the type of each answer and plot frequencies in Figure 5.6. Most questions ask about people (human), but with a broad diversity among other types.

These two breakdowns show that QB is topically and answer-wise diverse. To QB aficionados this is unsurprising; the primary educational goal of QB is to encourage students to improve their mastery over wide ranges of knowledge. We now turn to details about the gameplay dataset.

5.3.3 Gameplay Records: Recording Humans Playing Quizbowl Online

Like the 2002 TREC QA track (Voorhees, 2004), SQuAD 2.0 (Rajpurkar et al., 2018), and NQ (Kwiatkowski et al., 2019), deciding when *not* to answer is crucial to playing QB. Unlike these tasks though, deciding when to answer is not just model calibration or triggering, but also should reflect the opponent’s behavior (Billings et al., 1998). To address this, we use gameplay data (Table 5.2) which contain records of quizbowlers playing questions from prior tournaments: words in each question were revealed one-by-one until the player guessed the question’s answer. We use these records as (1) training data so that models can learn to imitate an oracle buzzing policy (Coates et al., 2008; Ross and Bagnell, 2010; Ross et al., 2011) and (2) as human baselines for offline evaluations (§5.7).

Like Mandel et al. (2014), gameplay records both simulate humans for training and evaluating policies. To simulate play against a human, we see which agent—human or machine—first switches from the wait action to the buzz action. For example, in Table 5.2 the user correctly guessed “Atlanta” at word forty-seven. If an agent played against this player they would need to answer correctly before word

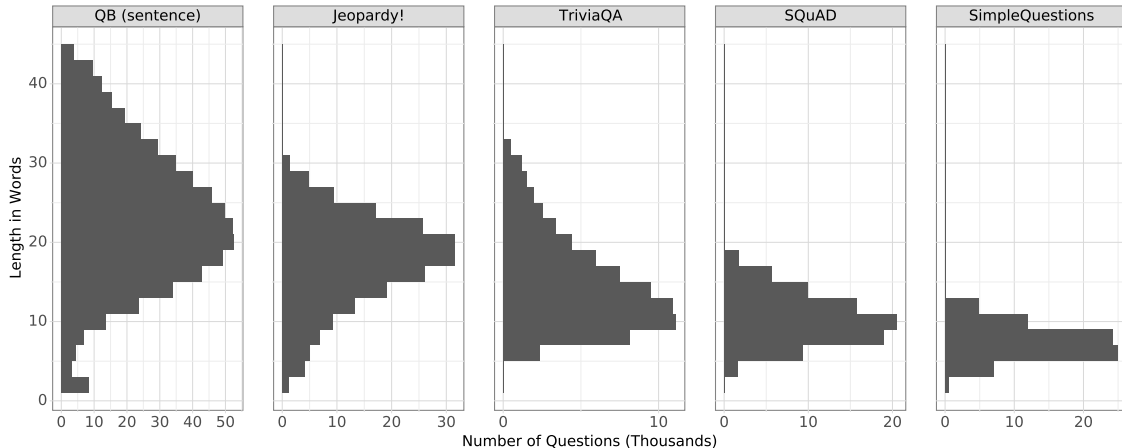


Figure 5.4: Size of question answering datasets. Questions in the QANTA dataset have longer sentences than any other dataset. The instances from SimpleQuestions, SQuAD, and TriviaQA are comparatively short, which makes it less likely that they are as diverse as QB or Jeopardy!. For each dataset, we compare the lengths of questions rather than paired context paragraphs; to avoid the histogram being overly skewed we remove the top 5% of examples by length from each dataset.

forty-seven to win. In all but one outcome, replaying the human record exactly recreates a live face-off. When a machine incorrectly buzzes first we lack what the human would ultimately guess, so we assume their guess would have been correct since skilled players almost always answer correctly by the end of the question. During training, these data help agents learn optimal buzzing policies based on their own uncertainty, the questions, and their opponents’ history (He et al., 2016b).¹⁹

With this data, we compute how models would fare against human players individually, players partitioned by skill, and in expectation (§5.7.1). In contrast to this strategy, crowdsourced tasks (e.g., SQuAD) often use the accuracy of a single annotator to represent human performance, but this is problematic as it collapses the distribution of human ability to a single crowd-worker and does not accurately reflect a task’s upper bound compared to multiple annotation (Nangia and Bowman, 2019; Kwiatkowski et al., 2019). In the gameplay data, we have ample data with which to robustly estimate average and sub-group human skill; for example, 90,611 of the 131,075 questions have been played at least five times. This wealth of gameplay data is one aspect of QB’s strength for comparing humans and machines.

An additional aspect unique to trivia games is that participants are intrinsically motivated experts. Compensation—i.e., extrinsic motivation—in crowdsourc-

¹⁹This work significantly expands the number of player-question records. We also make the setting significantly harder by not restricting questions to only the most frequently asked about answers (1K versus 24K). Finally, we create a new evaluation procedure (§5.7.1) that better estimates how models fare in the real-world versus human players. The first version of the gameplay dataset and models was introduced in:

He He, Jordan Boyd-Graber, and Hal Daumé III. **Opponent Modeling in Deep Reinforcement Learning**. *International Conference on Machine Learning*, 2016.

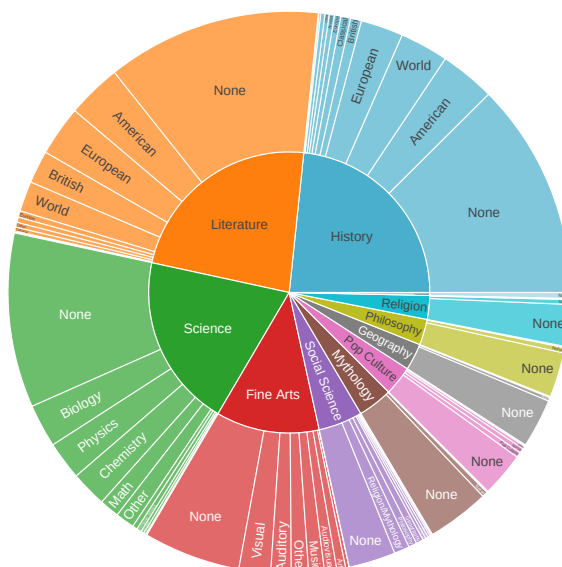


Figure 5.5: Questions in QB cover most if not all academic topics taught in school such as history, literature, science, the fine arts, and social sciences. Even within a single category, questions cover a range of topics. Topically, the dataset is biased towards American and European topics in literature and history.

ing is notoriously difficult. If they feel underpaid, workers do not give their best effort (Gneezy and Rustichini, 2000), and increasing pay does not always translate to quality (Mason and Watts, 2009). In light of this, Mason and Watts (2009) recommend intrinsic motivation, a proven motivator for annotating images (von Ahn and Dabbish, 2004) and protein folding (Cooper et al., 2010). Second, although multiple non-expert annotations can approach gold standard annotation, experts are better participants when available (Snow et al., 2008). Thus, other tasks may understate human performance with crowdworkers lacking proper incentives or skills.

Good quizbowlers are both accurate and quick. To measure skill, we compute and plot in Figure 5.7 the joint distribution of average player accuracy and buzzing position (percent of the question revealed). The ideal player would have a low average buzzing position (early guesser) and high accuracy; thus, the best players reside in the upper left region. On average, players buzzes with 65% of the question shown with 60% accuracy (Figure 5.7). Although there are other factoid QA and—more specifically—trivia datasets, QB is the first and only dataset with a large dataset of gameplay records which allows us to train models and run offline benchmarks.

5.3.4 Preprocessing

Before moving to model development, we describe necessary preprocessing to eliminate answer ambiguity, pair questions to gameplay data, and creating dataset folds that enable independent yet coordinated training of distinct guessing and buzzing models. Preprocessing is covered in significantly more detail in Appendix C.1.

Date	Thu Oct 29 2015 08:55:37 GMT-0400 (EDT)
UID	9e7f7dde8fdac32b18ed3a09d058fe85d1798fe7
QID	5476992dea23cca90550b622
Position	47
Guess	atlanta
Result	True
Question text	This Arcadian wounded a creature sent to punish Oeneus for improperly worshipping Artemis and killed the centaurs Rhaecus and Hylaeus...

Table 5.2: An entry from the gameplay dataset where the player correctly guesses “Atlanta” at word 47. The entry QID matches with the `PROTO_ID` field in the question dataset where additional information is stored such as the source tournament and year.

Dataset Folds The goal of the folds in the QANTA dataset is to standardize the training and evaluation of models for the guessing and buzzing sub-tasks. Towards this goal, we sub-divide the QANTA dataset by sub-task and standard machine learning folds (e.g., training, development, and test). We create the standard machine learning folds by partitioning the data according to tournament type and year. To increase the quality of evaluation questions, we only include questions from championship level tournaments in the development and test folds.²² To derive the final folds, we temporally divide the data (Arlot and Celisse, 2010) so that only (championship) questions from 2015 and onward are used in evaluation folds.

The subdivision by task simultaneously addresses the issue that some questions lack gameplay data (thus are not helpful for buzzer training) and partitioning the data so that the buzzer calibrates against questions unseen during training (details in Appendix C.1.3). Table 5.3 shows the size of each sub-fold; unassigned questions correspond to those where the answer to page matching process failed. Finally, hundreds of new QB questions are created every year which provides an opportunity for continually adding new training questions and replacing outdated test questions. Ultimately, this may help temper overconfidence in the generalization of models (Patel et al., 2008b) since we expect there to be covariate shift, prior probability shift, and domain shift in the data (Quionero-Candela et al., 2009) as questions evolve to reflect modern events.

The QANTA datasets, a copy of the Wikipedia data used, intermediate artifacts, and other related datasets are available at <http://datasets.qanta.org>.

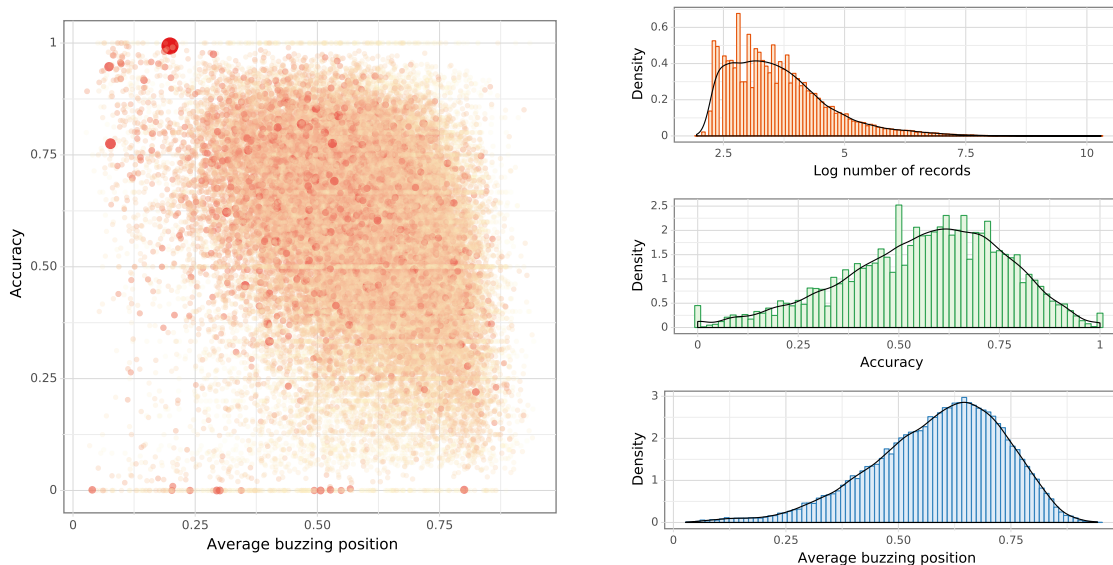


Figure 5.7: Left: each protobowl user is represented by a dot, positioned by average accuracy and buzzing position; size and color indicate the number of questions answered by each user. Right: distribution of number of questions answered, accuracy, and buzzing position of all users. An average player buzzes with 65% of the question shown, and achieves about 60% accuracy.

5.4 Deciding When and What to Answer

One could imagine many machine learning models for playing QB: an end-to-end reinforcement learning model or a heavily pipelined model that determines category, answer type, answer, and decides when to buzz. Without making any value judgment on the *right* answer, our approach divides the task into two subsystems: **guessing** and **buzzing** (Figure 5.8). This approach mirrors IBM Watson’s²³ two model design (Ferrucci et al., 2010; Tesauro et al., 2013). The first model answers questions, and the second decides when to buzz. Dividing a larger task into sub-tasks is common throughout machine learning, particularly when the second is making a prediction based on the first’s prediction. For example, this design pattern is used in object detection (Girshick et al., 2014, generate bounding box candidates then classify them), entity linking (Ling et al., 2015, generate candidate mentions and then disambiguate them to knowledge base entries), and confidence estimation for automatic speech recognition systems (Kalgaonkar et al., 2015). In our factorization, guessing is based solely on question text. At each time step (word), the guessing model outputs its best guess, and the buzzing model determines whether to buzz or wait based on the guesser’s confidence and features derived from the game state. This factorization cleanly reduces the guesser to question answering while framing

²²We use questions from ACF Regionals, ACF Nationals, ACF Fall, PACE NSC, and NASAT from 2015 onward for development and test sets.

²³In Watson, the second system also determines wagers on Daily Doubles, wagers on Final Jeopardy, and chooses the next question (e.g., history for \$500)

Fold	Number of Questions
train + guess	96,221
train + buzz	16,706
dev + guess	1,055
dev + buzz	1,161
test + guess	2,151
test + buzz	1,953
unassigned	13,602
All	132,849

Table 5.3: We assign each question in our dataset to either the train, development, or test fold. Questions in the development and test folds come from national championship tournaments which typically have the highest quality questions. The development and test folds are temporally separated from the train and development folds to avoid leakage. Questions in each fold are assigned a “guess” or “buzz” association depending on if they have gameplay data. Unassigned refers to questions for which we could not map their answer strings to Wikipedia titles or there did not exist an appropriate page to match to.

the buzzer as a cost-sensitive confidence calibrator.

This division of modeling labor makes it significantly easier to train the buzzer as a learned calibrator of the guesser’s softmax classifier predictions. This is crucial since the probabilities in neural softmax classifiers are unreliable (Guo et al., 2017). Like how we train a calibration model (buzzer) over a classifier (guesser), Corbière et al. (2019) train a calibration model on top of an image classification model which is a more effective approach in high dimensional spaces compared to nearest-neighbor based confidence measures (Jiang et al., 2018). However, not all buzzing errors are equal in severity; thus, part of the buzzer’s challenge is in incorporating cost-sensitive classification. By partitioning model responsibilities into separate guessing and buzzing models, we can mitigate the calibration-based drawbacks of neural softmax classifiers while naturally using gameplay data for cost-sensitive decision-making.

Machines playing QB by guessing and buzzing semi-independently is also convenient from an engineering perspective: it simplifies model training and is easier to debug. More importantly, it allows us and subsequent researchers to focus on a sub-task of their choosing or the task as a whole. If you are interested in only question answering, focus on the guesser. If you are interested in multi-agent cooperation or confidence estimation, focus on the buzzer. Following the discussion of our guessing (§5.5) and buzzing (§5.6) systems we describe our evaluations and results in Section 5.7.1. Section 5.8 summarizes the outcomes of our live, in-person, exhibition matches against some of the best trivia players in the world.

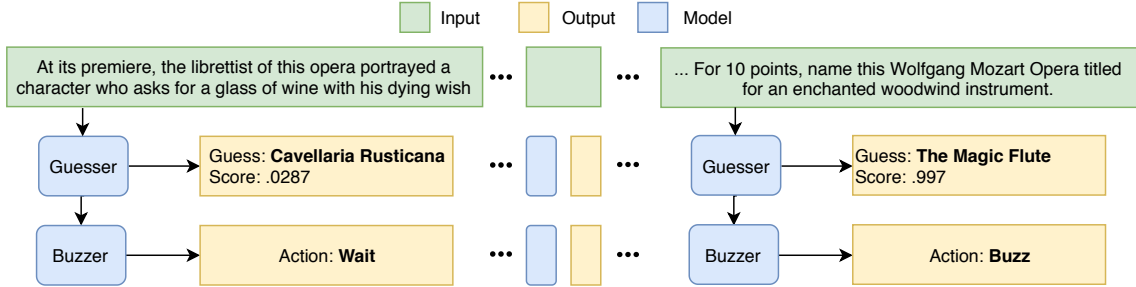


Figure 5.8: The QANTA framework for playing Quiz Bowl with semi-independent guesser and buzzer models. After each word in the input is revealed the guesser model outputs its best guesses. The buzzer uses these in combination with positional and gameplay features to decide whether to take the buzz or wait action. The guesser is trained as a question answering system that provides guesses given the input text seen so far. Buzzers take on dual roles as calibrators of the guesser confidence scores and cost-sensitive decision classifiers by using the guesser’s score, positional features, and human gameplay data.

5.5 Guessing QB Answers

Guessing answers to questions is a factoid question answering task and the first step towards our models playing QB (Figure 5.8). We frame the question answering sub-task in QB as high dimensional multi-class classification over Wikipedia entities (i.e., answers are entities defined by distinct Wikipedia pages). This section describes three families of question answering models: information retrieval models (§5.5.1), linear models (§5.5.2), and neural models (§5.5.3). Despite distinct differences, these approaches share a common structure: create a vector representation \mathbf{x} of the input question, create a vector representation for each candidate answer \mathbf{a}_i , and then return the answer A_i corresponding to $\arg \max_i f(\mathbf{x}, \mathbf{a}_i)$ where f is some similarity function.²⁴

5.5.1 Explicit Pattern Matching with Information Retrieval

The first model family we discuss are traditional information retrieval (IR) models based on the vector space model (Salton et al., 1975). Vector space models are particularly effective when term overlap is a useful signal—as in factoid QB (Lewis et al., 2020a). For example, although early clues avoid keyword usage, giveaways often include terms like “Wolfgang Mozart” and “Tamino” that make reaching an answer easier. Consequently, our vector space IR prove to be a strong baseline (§5.7.1).

To frame this as an IR search problem, we treat guessing as document retrieval. Input questions are search queries and embedded into a TF-IDF (Jones, 1972; Rajaraman and Ullman, 2011) vector \mathbf{x} . For each answer $A_i \in \mathcal{A}_{train}$ in the QB training data, we concatenate all training questions with that answer into a document D_i

²⁴For brevity and clarity, we omit bias terms.

embedded as \mathbf{a}_i into the same vector space.²⁵ The textual similarity function f is Okapi BM25 (Robertson and Walker, 1994) and scores answers \mathbf{a}_i against \mathbf{x} . During inference, we return the answer A_i of the highest scoring document D_i . We implement our model using Apache Lucene and Elastic Search (Gormley and Tong, 2015).

However, the IR model’s reliance on pattern matching often fails early in the question. For example, in the first sentence from Figure 5.1 the author intentionally avoids keywords (“a character who asks for a glass of wine with his dying wish”). Purely traditional IR methods, while effective, are limited since they rely on keywords and cannot “soft match” terms semantically. Thus, we move on to machine learning methods that address some of these shortcomings.

5.5.2 Trainable Pattern Matching with Linear Models

In addition to the IR model, we also test a linear model baseline that reduces multi-class classification to one-versus-all binary classification. While an IR model derives term weights from corpus statistics and a hand-crafted weighting scheme, a one-versus-all linear model with one-hot term features \mathbf{x} finds term weights that maximize the probability of the correct binary prediction for each answer. The input features \mathbf{x} are derived from a combination of sparse n-grams and skip-grams.²⁶ Since the number of classes is too high for standard one-versus-all multi-class classification,²⁷ we instead use a logarithmic time one-versus-all model (Agarwal et al., 2014; Daumé et al., 2017). However, this model is limited since it only considers linear relationships between n-gram terms, the model uses—at best—local word order, and the sparse representation does not take advantage of the distributional hypothesis (Harris, 1954). Next we describe neural models that use more sophisticated forms of representation and composition to address these shortcomings.

5.5.3 Neural Network Models

The final family of methods we consider for QB question answering are neural methods. We describe the shared components of the neural models (e.g., general architectures and training details) and compare their composition functions.

In our model (Figure 5.9), we follow a widely used architecture in NLP to embed words independently in a vector space, contextualize their representations, temporally reduce representations, and then classify with a softmax layer (Collobert and Weston, 2008). The first component of the model embeds question q with k tokens into m -dimensional representations $\mathbf{w} = [\mathbf{w}_1, \dots, \mathbf{w}_k]$. Next, a function $c(\cdot) : \mathbb{R}^{k \times m} \rightarrow \mathbb{R}^{k \times l}$ contextualizes words as l -dimensional embeddings $\mathbf{v} = [\mathbf{v}_1, \dots, \mathbf{v}_k] = c(\mathbf{w})$. Since this is still a variable length sequence of representations and the classifier requires a fixed size representation, we use a reducer

²⁵We also tested, one document per training example, different values for BM25 coefficients, and the default Lucene practical scoring function.

²⁶The order of n-grams and skip-grams was determined by hyper parameter search

²⁷There are approximately 25,000 distinct answers.

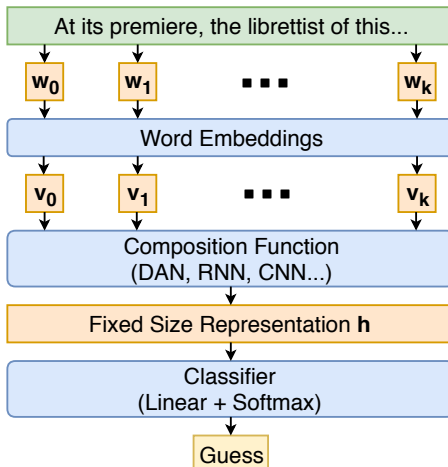


Figure 5.9: All our neural models feed their input to an embedding function, then a composition function, and finally a classification function. The primary variation across our models is the choice of composition function used to compute a fixed, example-level representation from its variable length input.

$r(\cdot) : \mathbb{R}^{k \times m} \rightarrow \mathbb{R}^n$ to derive an n -dimensional dense feature vector $\mathbf{x} = r(\mathbf{v})$. We call specific pairs of contextualizers and reducers *composition functions* (§2.4.3). The final model component—the classifier—computes logit scores $s_i = \mathbf{x}^T \cdot \mathbf{a}_i$ as the dot product between the features \mathbf{x} and trainable answer embeddings \mathbf{a}_i . From this, we use the softmax to compute a probability distribution

$$\mathbf{p} = \text{softmax}(\mathbf{s}) = \frac{\exp(\mathbf{s})}{\sum_{i=1}^k \exp(s_i)} \quad (5.1)$$

over answers and train the model with the cross entropy loss

$$\mathcal{L} = \sum_i^k y_i \log(\hat{p}_i) \quad (5.2)$$

where $y_i = 1$ for the true answer and $y_i = 0$ otherwise. In our experiments, we evaluate three classes of composition functions (i.e., contextualizer-reducer pairs): unordered composition with deep averaging networks (Iyyer et al., 2015), recurrent network-based composition (Elman, 1990; Hochreiter and Schmidhuber, 1997; Palangi et al., 2016; Cho et al., 2014a), and transformer-based composition (Vaswani et al., 2017; Devlin et al., 2018).

Unordered Composition with Deep Averaging Networks Our first (unordered) neural composition function is the deep averaging network (DAN). We introduced DANs as a simple, effective, and efficient method for QB question answering.²⁸ Despite their disregard of word order, DANs are competitive with more

²⁸This work has new experiments comparing new composition functions and focuses on incorporating additional data. The DAN first was introduced in:

sophisticated models on classification tasks such as sentiment analysis (Iyyer et al., 2015). Although there are cases where word order and syntax matter, many questions are answerable using only key phrases. For example, predicting the mostly likely answer to the bag of words “inventor, relativity, special, general” is easy; they are strongly associated with Albert Einstein.

All composition functions—such as DANs—are fully described by the choice of contextualizer and reducer. In DANs, the contextualizer c is the identity function, and the reducer is broken into two components. First, the DAN averages word embeddings \mathbf{v} to create an initial hidden state

$$\mathbf{h}_0 = \frac{1}{k} \sum_{i=1}^k \mathbf{v}_i. \quad (5.3)$$

The final fixed-size representation $\mathbf{x} = \mathbf{h}_z$ is computed with z feed-forward layers through the recurrence

$$\mathbf{h}_i = \text{GELU}(\mathbf{W}_i \cdot \mathbf{h}_{i-1} + \mathbf{b}_i) \quad (5.4)$$

where \mathbf{W}_i and \mathbf{b}_i are parameters of the model and GELU is the Gaussian Error Linear Unit (Hendrycks and Gimpel, 2017b). Although DANs are not the most accurate model, they are an attractive trade-off between accuracy and computation cost.

Ordered Composition In contrast to DANs, order-aware models like RNNs, LSTMs, and GRUs can model long range dependencies in supervised tasks (Linzen et al., 2016). Since all these models belong to the family of recurrent models, we choose one variant to describe in terms of its associated contextualizer and reducer.²⁹ In our model, the composition function

$$c(\mathbf{v}) = \text{GRU}(\mathbf{v}) \quad (5.5)$$

is a multi-layer, bi-directional GRU (Cho et al., 2014a). The reducer

$$r(\mathbf{v}) = [\mathbf{v}_k^{(\text{forward})}; \mathbf{v}_0^{(\text{backward})}] \quad (5.6)$$

concatenates the final layer’s forward and backward hidden states. Combined, this forms the first ordered composition we test.

Transformer models, however, better represent context at the cost of complexity (Vaswani et al., 2017; Devlin et al., 2018). Specifically, we input the CLS token, question, and SEP token to uncased BERT-BASE. Thus, the contextualizer

$$c(\mathbf{v}) = \text{BERT}(\mathbf{v}) \quad (5.7)$$

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. **Deep Unordered Composition Rivals Syntactic Methods for Text Classification**. *Association for Computational Linguistics*, 2015.

²⁹Hyper parameter optimization indicated that GRU networks were the most accurate recurrent model.

is simply BERT and the reducer

$$r(\mathbf{v}) = \frac{1}{k} \sum_{i=1}^k \mathbf{v}_i \tag{5.8}$$

is the average of the output states from the final layer associated with the question’s wordpiece tokens.³⁰ Next, we move on from model descriptions to training specifics.

Training Details In standard QA tasks, training over full questions is standard, but with QB’s incremental setup this results in less accurate predictions. If the example is the complete text, the model ignores difficult clues to focus on the “easy” part of the question, preventing learning from “hard” clues. Instead of each training example being one question, we use each of a question’s sentences as a single training example. While this training scheme carries the downside that models may not learn long-range dependencies across sentences, the empirical accuracy improvement outweighs the disadvantages. In addition to these two approaches, we also tested variable-length, but did not observe an improvement over the sentence-based training scheme.³¹

In non-transformer models we use 300-dimensional word embeddings initialized with GLoVe for words in the vocabulary and randomly initialized embeddings otherwise.³² We regularize these models with dropout (Srivastava et al., 2014) and batch normalization (Ioffe and Szegedy, 2015). Loss functions were optimized with ADAM (Kingma and Ba, 2015) and models were trained with early stopping, and learning rate annealing. All neural models were implemented in PyTorch (Paszke et al., 2019) and AllenNLP (Gardner et al., 2018).

We optimize hyper parameters by running each setting once and record the parameter settings corresponding to the top development set accuracy. The models with the best parameters are run an (additional) five times to create estimates of variance for each tracked metric (§5.7.1).

Although not exhaustive, these models are strong baselines for the question answering component of QB. Section 5.10 identifies areas for future modeling work; throughout the rest of this work however we focus on completing a description of our approach to playing QB by combining guessers and buzzer (§5.6). Following this we describe how we evaluate these systems independently (§5.7.1), jointly (§5.7.3), offline, and in live events (§5.8).

5.6 Buzzing

Winning in QB requires answering accurately with as little information as possible. It is crucial—for humans and computers alike—to accurately measure

³⁰We also tested using the CLS token with worse results.

³¹Variable length training creates k training examples from a question comprised of k sentences. Each example includes the text from the start position up to and including sentence k .

³²Randomly initialized embeddings use a normal distribution with mean zero and standard deviation one.

their confidence and buzz as early as possible without being overly aggressive. The first part of our system, the guesser, optimizes for guessing accuracy; the second part, the buzzer, focuses on deciding when to buzz. Since questions are revealed word-by-word the buzzer makes a binary decision at each word: buzz and answer with the current best guess, or wait for more clues.

The outcome of this action depends on the answers from both our guesser and the opponent.³³ To make this clear, we review the game’s mechanics. If we buzz with the correct answer before the opponent can do so, we win 10 points; but if we buzz with an incorrect answer, we lose 5 points immediately, and since we cannot buzz again, the opponent can wait till the end of the question to answer, which might cost us 10 extra points in the competition. Thus, it should be clear that “a wrong answer in Quizbowl is extraordinarily costly” (Jennings, 2006, p. 93).

Before we discuss our strategy to buzzing, consider a buzzer with perfect knowledge of whether the guesser is correct or not, but does not know anything about the opponent: a *locally optimal* buzzer. This buzzer would buzz as soon as the guesser gets the answer correct. A stronger buzzer exists: an omnipotent buzzer with perfect knowledge of what the opponent will do; it would exploit the opponent’s weaknesses: delay buzzing whenever an opponent might err. The agent would then get a higher relative reward: once from the opponent’s mistake and then for getting it correct.

The buzzer we develop in this chapter targets a locally optimal strategy: we focus on predicting the correctness of the guesser and do not model the opponent. This buzzer is effective: it both defeats players in our gameplay dataset (§5.3.3) and playing against real human players (§5.8). The opponent modeling extension has been explored by previous work, and we discuss it in Section 5.9.

5.6.1 A Classification Approach to Buzzing

Given the initial formulation of buzzing as a MDP (§5.2.5), it would be natural to learn the task with reinforcement learning using the final score; however, we instead use a convenient reduction to binary classification. Since we can compute the optimal buzzing position easily as opposed to with expensive rollouts, we can reduce the problem to classification (Lagoudakis and Parr, 2003). At each time step, the model looks at the sequence of guesses that the guesser has generated so far, and makes a binary decision of whether to buzz or to wait. Under the locally optimal assumption, the ground truth action at each time step equals the correctness of the top guess: it should buzz if and only if the current top guess is correct. Another view of this process is that the buzzer is learning to imitate the oracle buzzing policy from the ground truth actions (Coates et al., 2008; Ross and Bagnell, 2010; Ross et al., 2011). Alternatively, the buzzer can also be seen as an uncertainty estimator (Hendrycks and Gimpel, 2017a) of the guesser.

The guesses create a distribution over all possible answers. If this distribution

³³We use point values from the typical American format of the game. The exact values are unimportant, as they change the particulars of strategy but not the approach.

faithfully reflects the uncertainty of guesses, the buzzer could be a simple “if–then” rule: buzz as soon as the guesser probability for any guess gets over a certain threshold. This *threshold* system is our first baseline, and we tune the threshold value on a held-out dataset.

However, this does not work because the confidence of neural models is ill-calibrated (Guo et al., 2017; Feng et al., 2018) and worsens with domain shift (Kamath et al., 2020). Our neural network guesser often outputs a long tail distribution over answers concentrated on the top few guesses, and the confidence score of the top guess is often higher than the actual uncertainty (the chance of being correct). To counter these issues, we extract features from the top ten guesser scores and train a classifier on top of them. Some important features include a normalized version of the top ten scores and the gap between them (Appendix C.1.5 lists all features).

There is also important temporal information; for example, if the guesser’s top prediction’s score steadily increases, this signals the guesser is certain about the top guess. Conversely, a fluctuating top prediction (the answer is Hope Diamond. . . no, I mean Parasaurlophus. . . no, I mean Tennis Court Oath) is a sign that perhaps the guesser is not that confident (regardless of the ostensible score). To capture this, we compare the current guesser scores with the previous time steps and extract features such as the change in the score associated with the current best guess, and whether the ranking of the current top guess changed in this time step.

To summarize, at each time step, we extract a feature vector, including current and temporal features, from the sequence of guesses generated by the guesser so far. We implement the classifier with both fully connected Multi-layer Perceptron (MLP) and with Recurrent Neural Network (RNN). The classifier outputs a score between zero and one indicating the estimated probability of buzzing. Following the locally optimal assumption, we use the correctness of the top guess as ground truth action: buzz if correct and wait if otherwise. We train the classifier with logistic regression; during testing, we buzz as soon as the buzzer outputs a score greater than 0.5. Both models are implemented in Chainer (Tokui et al., 2015); we use hidden size of 100, and LSTM as the recurrent architecture. We train the buzzer on the “buzzertrain” fold of the dataset, which does not overlap with the training set of the guesser, for twenty epochs with the Adam optimizer (Kingma and Ba, 2015). Both buzzers have test accuracy of above 80%, however, the classification accuracy does not directly translate into the buzzer’s performance as part of the pipeline, which we look at next.

5.7 Offline Evaluation

A core idea of this chapter is that the construction of QB questions lends itself to a fairer evaluation of both humans and machine QA models: to see who is better at answering questions, see who can answer the question first. However, this is often impractical during model development, especially if the questions are “new” (they have not been played by humans or computers). Moreover, a researcher might be uninterested in solving the buzzing problem. Offline evaluations where the

guesser and buzzer are evaluated independently with static data strikes a balance between ease of model development and faithfulness to QB’s format. To address this, Section 5.7.1 describes the metrics to compare offline model accuracy. Following an error analysis (§5.7.2), Section 5.7.3 evaluates buzzing models by replacing this oracle buzzer with trained buzzing models.

5.7.1 Evaluating the Guesser

Ideally, we would compare systems in a head-to-head competition where the model (or human) who correctly buzzed and answered the most questions would win (§5.8). However, this involves live play, necessitates a buzzing strategy, and complicates evaluation of the guesser in isolation. Intuitively though, a model that consistently buzzes correctly earlier in the question is better than a model that buzzes late in the question. In our evaluations, we use three metrics that reflect this intuition: accuracy early in the question, accuracy late in the question, and the expected probability of beating a human assuming an optimal buzzing strategy.

Accuracy-Based Evaluation The easiest and most common method for evaluating closed domain question answering methods is accuracy over all questions in the test set. We report two variants of this: (1) accuracy using the first sentence and (2) accuracy using the full question. While it is possible to answer some questions during the first sentence, it is the first and hardest position we can guarantee *could* be answered. Although we report accuracy on full questions, this metric is a minimum bar: the last clues are intentionally easy (§5.2.3). However, while start-of-question and end-of-question accuracy help development and comparison with other QA tasks, it is silent on human–computer comparison. We address this shortcoming next.

Expected Probability of Defeating Human Players While comparing when systems buzz is the gold standard, we lack gameplay records for all test set questions, and it is unreasonable to assume it is easy to obtain them. Instead, marginalize over empirical human gameplay to estimate the probability $\pi(t)$ that a human would have correctly answered a question by position t . Then, we combine this with model predictions and marginalize over t to obtain the expected probability of winning against an average player on an average gameplay question. A similar idea—to compute the expected probability of winning a heads up match—has also been used in machine translation (Bojar et al., 2013).

We compute the expected probability of winning (EW) in two steps. First, we compute the proportion of players

$$\pi(t) = 1 - \frac{N_t}{N}, \tag{5.9}$$

that have answered a question correctly by position t . N is the total number of question-player records and N_t is the number of question-player records where the

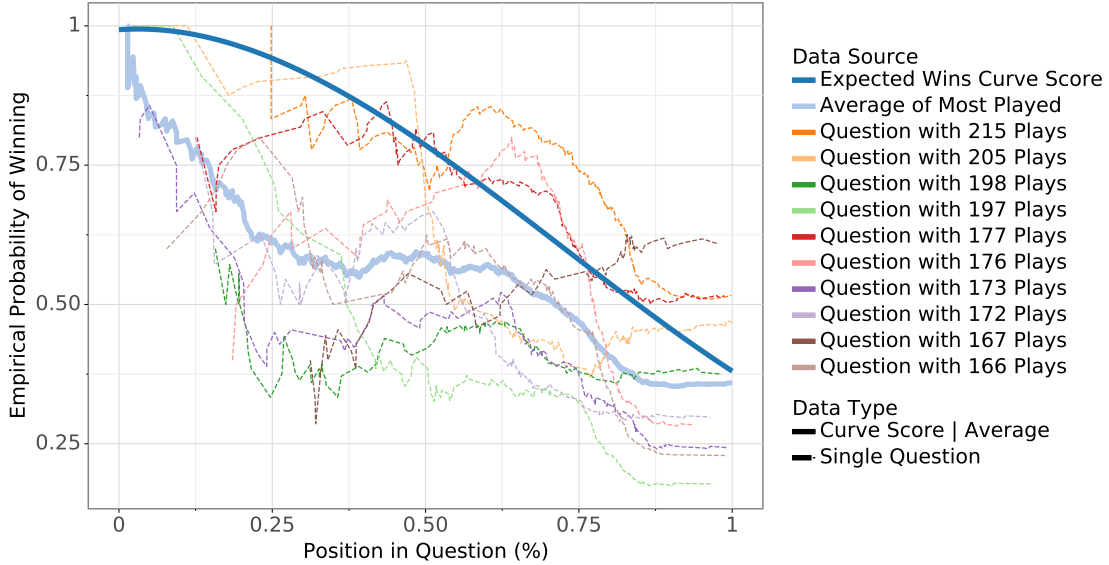


Figure 5.10: We plot the expected wins score with respect to buzzing position (solid dark blue). For the ten most played questions in the buzztest fold we show the empirical distribution for each individual question (dotted lines) and when aggregated together (solid light blue). Among the most played questions, expected wins over-rewards early buzzes, but appropriately rewards end-of-question buzzes.

player answered correctly by position t . We empirically estimate the expected probability of winning

$$\pi(t) = 0.0775t - 1.278t^2 + 0.588t^3 \quad (5.10)$$

from the gameplay data as a cubic polynomial (Figure 5.10). At $t = 0$, the potential payoff is at its highest since no one has answered the question. At $t = \infty$, the potential payoff is at its lowest; all the players who would have correctly answered the question already have. If the computer gets the question right at the end, it would only score points against opponents who did not know the answer at all or answered incorrectly earlier in the question.

EW marginalizes over all questions q and all positions j , and counts how many times model m produced a guess $g(m, q, j)$ that matched the answer of the question $a(q)$. Specifically we compute

$$\text{EW}(m) = \mathbb{E}_m [p_{\text{win}}] = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \sum_{j=1}^{\infty} \mathbb{1}[g(m, q, j) = a(q)] \pi(j), \quad (5.11)$$

where $\frac{1}{|\mathcal{Q}|}$ is the count of question–position records. The indicator function is exactly an oracle buzzer: it gives credit if and only if the answer is correct. However, this rewards models with unstable predictions; for example, a model would be rewarded twice for a sequence of predictions that were correct, wrong, and correct. We discourage this model behavior by using a *stable* variant of EW which only awards points if the current answer and all subsequent answers are correct. With this formalism, it is also straightforward to compute the expected winning probability for

Model	Accuracy (%)									
	Start				End				$\mathbb{E}[p_{\text{win}}]$	
	Dev		Test		Dev		Test			
	Top	Mean	Top	Mean	Top	Mean	Top	Mean	Dev	Test
Lin.	2.56	2.56±0.	1.58	1.58±0.	11.9	11.9±0.	9.25	9.25±0.	6.62	4.96
IR	9.48	9.48	6.23	6.23	62.2	62.2	54.5	54.5	45.8	38.8
DAN	10.7	10.4±0.3	8.28	7.88±0.3	60.0	59.1±0.9	51.0	51.4±1	42.6	35.5
RNN	10.5	9.46±0.7	7.86	7.78±0.4	52.3	51.8±1	46.4	45.9±0.9	27.6	23.3
BERT	12.5	11.1±0.8	9.34	9.49±0.3	53.4	55.0±0.9	47.0	48.8±0.9	36.6	31.6

Table 5.4: We compare several models by accuracy at start-of-question, end-of-question, and EW. In the table, models are sorted by start-of-question development set accuracy. Standard deviations for non-IR models are derived from five trials; standard deviation is not reported for the IR model since it is deterministic.

any guesser-buzzer combination by replacing the oracle buzzer (indicator function) with a function that equals one if the guess is correct and the buzzer yielded a “buzz” decision. We compare buzzers in Section 5.6, but now move to experimental results for the guesser.

Guesser Comparison Experiments We evaluate our guessers using start accuracy, end accuracy, and expected wins (Table 5.4). All models struggle at the start of the question with the best accuracy at only 9.34%. This is unsurprising: the first sentence contains the most difficult clues and is difficult for even the best human players. Models fare significantly better near the end of the question with giveaway clues. However, even the best model’s 54.5% accuracy leaves much room for future work.

While the BERT model has the best early-question accuracy, it lags behind the IR and DAN for end of question accuracy. We suspect that order-aware models over-emphasize less important parts of the question; additionally, the gap between sentence training and full question inference advantages models that did not need to learn an aggregation over longer sequences. This pattern is also reflected in the EW scores; BERT—as expected—outperforms the RNN model. Finally, across accuracy and EW we see substantial drops between the development and test sets, which suggests overfitting. Next, we investigate the errors models make.

5.7.2 Identifying Sources of Error

This section identifies and characterizes several failure modes of our models. First, we compare the predictions of blackbox neural models and an IR model—an explicit pattern matcher. Following this, we identify data skew towards popular answers as a major source of error for less popular answers. Lastly, we manually break down the test errors of one model.

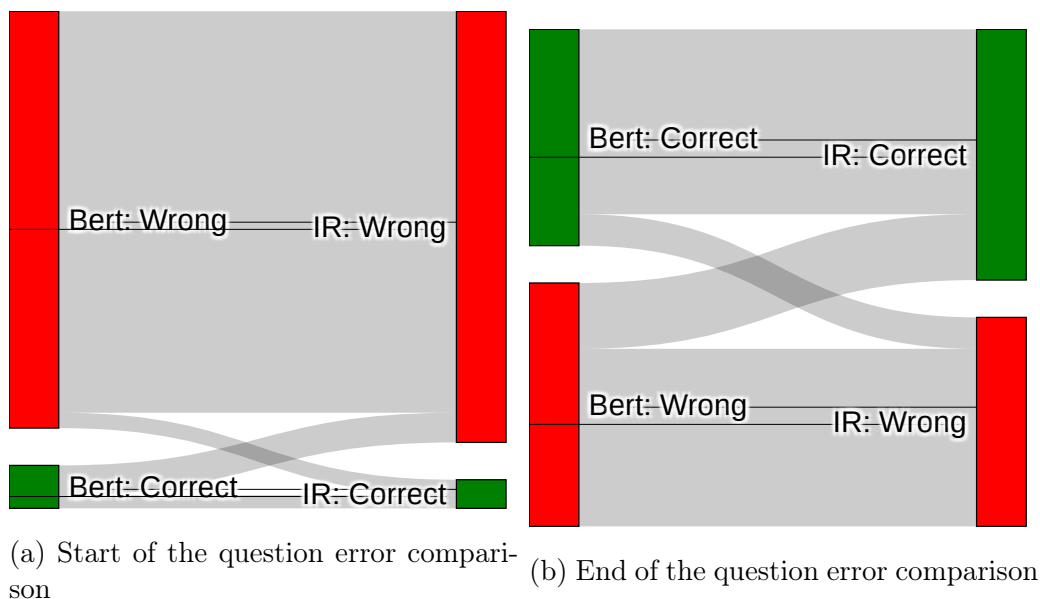


Figure 5.11: The BERT and IR models are mostly wrong or correct on the same subset of questions. At the end of the question, most of the questions the BERT model is correct on, the IR model is also correct on.

Behavioral Comparison of Neural and IR Models One way to analyze black-box models like neural networks is to compare their predictions to better understood models like the IR model. If their predictions—and thus exterior behavior—are similar to a better understood model it suggests that they may operate similarly. Figure 5.11 shows that even the BERT and IR models are correct and wrong on many of the same examples at end-of-question. Since one model—the IR model—is an explicit pattern matcher this hints that neural QB models learn to be pattern matcher as hinted by other work (Jia and Liang, 2017; Rajpurkar et al., 2018; Feng et al., 2018).

Next we investigate this pattern matching hypothesis at the instance-level. For our instance-level analysis we sample examples of correct and incorrect predictions. First we randomly sample a test question that all models answer correctly after the first sentence (Figure 5.12). This particular example has similar phrasing to a training example (“A holder of this title commissioned... miniatures”) so it is unsurprising that all models get it right.

In our second analysis, we focus on a specific answer (Turbulence) and its twenty-seven training questions. Figure 5.13 shows a sample question for this answer that the RNN model answered correctly but that IR model did not. The most frequent words in the training data for this answer are “phenomenon” (twenty-three times), “model” (seventeen times), “equation” (thirteen times), “numerically” (once), and “tensor” (once). In this analysis we removed or substituted these word with synonyms and then checked if the model’s prediction was the same.

Substituting words in this question shows that the model is over-reliant on specific terms. After removing the term “phenomenon,” the model changed its answer

Test Question (first sentence):

A holder of this title commissioned a set of miniatures to accompany the story collection Tales of a Parrot.

Training Question (matched fragment):

A holder of this title commissioned Abd al-Samad to work on miniatures for books such as the Tutinama and the Hamzanama.

Answer: Mughal Emperors

Figure 5.12: A test question that was answered correctly by all models after the first sentence; a normally very difficult task for both humans and machines. A very similar training example allows all models to answer the question through trivial pattern matching.

Test Question (first sentence):

This phenomenon is resolved without the help of a theoretical model in costly DNS methods, which numerically solve for the rank-2 tensor appearing in the RANS equations.

Answer: Turbulence Score (RNN): .0113

Synonym Attacks: phenomenon → event, model → representation

Figure 5.13: Only the RNN model answers this question correctly. To test the robustness of the model to semantically equivalent input modifications, we use SEARS-based (Ribeiro et al., 2018) synonym attacks and cause the model prediction to become incorrect. Although this exposes a flaw of the model, it is also likely that the low confidence score would likely lead a buzzer model to abstain; this highlights one benefit of implicitly incorporating confidence estimation into the evaluation.

to Ising model (a mathematical model of ferromagnetism). If we instead substitute the term with synonyms such as “occurrence”, “event”, and “observable event” the answers are still incorrect. Similarly, if “model” is replaced by “representation” the RNN also makes incorrect predictions. At least for this question, the model is not robust to these semantics-preserving modifications (Ribeiro et al., 2018). Next we move to aggregate error analysis.

Errors Caused by Data Sparsity For many test set answers, scarcity of training data is a significant source of error. Most egregiously, 17.9% of test questions have zero corresponding training examples. Beyond these questions, many more answers have few training examples. While some topics are frequently asked about, one goal of question writers is to introduce new topics for students to learn from. For example, although physics is a common general topic, Electromagnetism has only been an answer to one QB question. The distribution of training examples per unique answers is skewed (Figure 5.14), and countries—like Japan—are asked about much more frequently. Unsurprisingly, plotting the number of training examples per test question answer versus model accuracy shows significant drops in accuracy for about

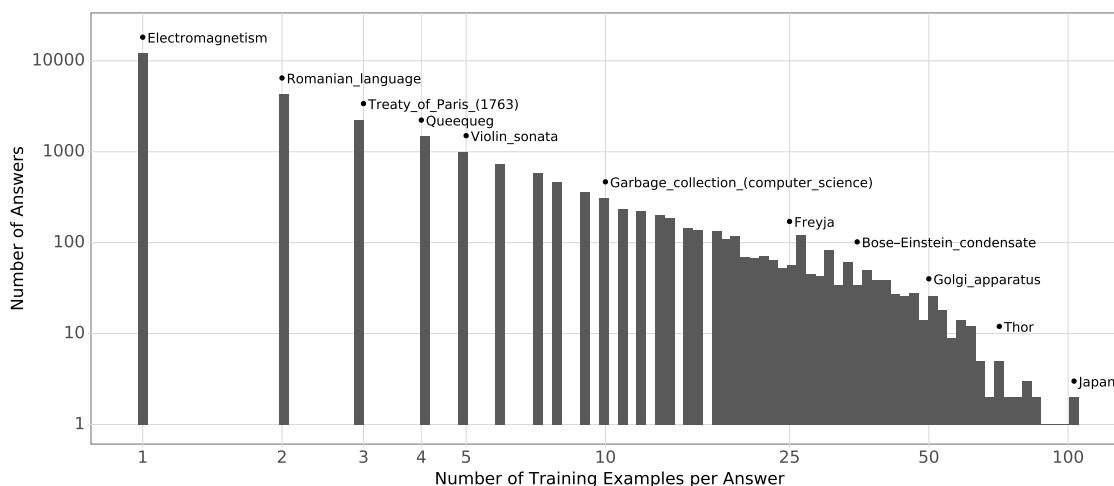


Figure 5.14: The distribution of training examples per unique answer is heavily skewed. The most frequent answer (Japan) occurs about 100 times. Nearly half of the questions have one training example and just over sixty percent have either one or two training examples.

half of the test questions (Figure 5.15).

Error Breakdown We conclude our error analysis by inspecting and breaking down the errors made by the RNN model at the start and end of questions. Of the 2,151 questions in the test set, 386 have zero training examples leaving 1,765 questions that are answerable by our models. Of the remaining questions, the RNN answers 1,540 incorrectly after the first sentence and 481 at the end of the question. To avoid errors likely due to data scarcity, we only look to questions with at least 25 training examples; the number of errors on this subset at the start and the end of the question is 289 and 36. Table 5.5 lists reasons for model errors on a random sample of 50 errors from the start of the question and all 36 errors from the end of the question.

The predominant sources of error are when the model predicts the correct answer type (e.g., person, country, place), but chooses the incorrect member of that type. This accounts for errors such as choosing the wrong person, country, place, or event. The RNN especially confuses countries; for example, in Figure 5.16, it confuses the Spain and the United States, the parties to the Adam Onis Treaty. The relative absence of incorrect answer type errors at the end of questions may be attributable to the tendency of late clues including the answer type (such as “name this country...”).

In the process of manual error breakdown we also found five annotation errors where the assigned Wikipedia answer did not match the true answer. This low number of errors further validates the robustness of our answer mapping process.

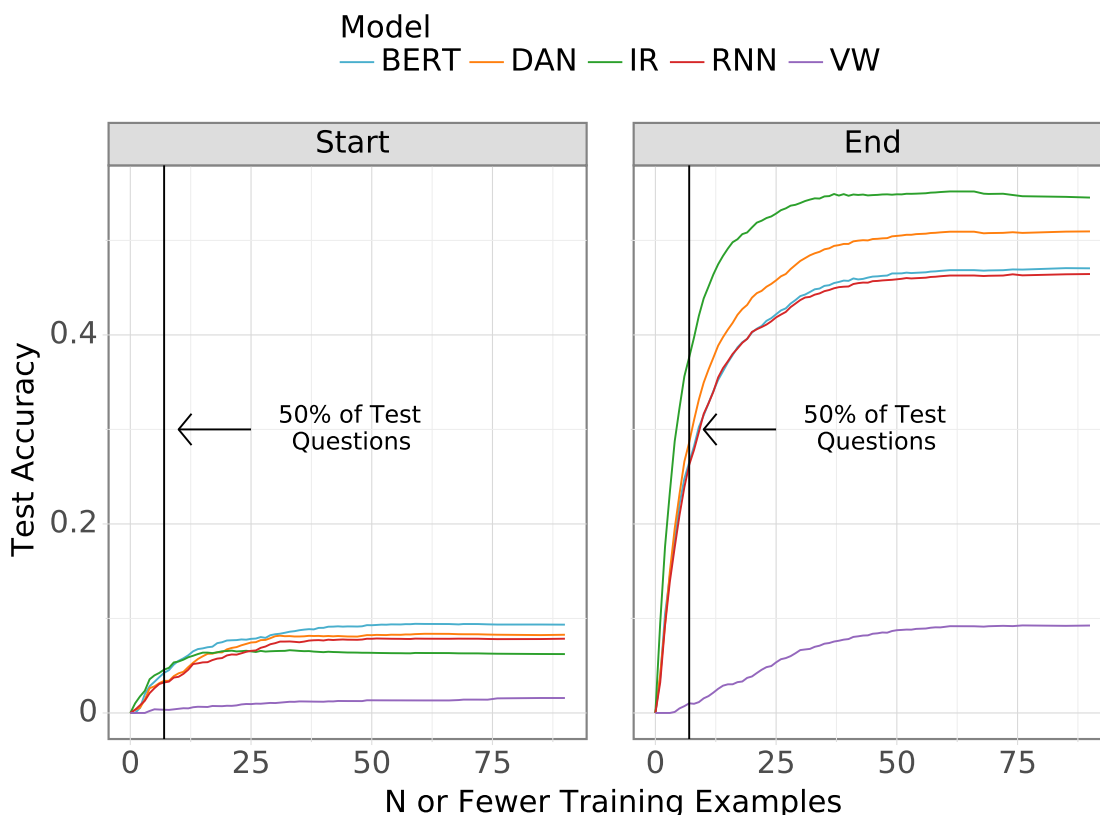


Figure 5.15: The more an answer is asked in the training set, the easier it is for all models, both at the start and end of the question. This is a significant source of errors since accuracy on at least 50% of test questions—those with seven or less training examples—is significantly lower for all models.

5.7.3 Evaluating the Buzzer

We first evaluate our buzzer against the locally optimal buzzer which buzzes as soon as the guesser gets the answer correct. However, this can be overly ambitious and unrealistic since the guesser is not perfectly stable: it can get the answer correct by chance, then vacillate between several candidates before settling down to the correct answer. To account for this instability, we find the first position that the guesser stabilizes to the correct answer and set it as the optimal buzzing position. In other words, we compare against the optimal buzzer used in the definition of stable EW score. To be exact, we start at the last position that the guess is correct, go backwards until the guess is incorrect and consider this the locally optimal buzzing position; we set the ground truth to all positions before this to zero, and all positions after it to one.

We use the same guesser (RNN) in combination with different buzzers (Table 5.6), and quantitatively compare their expected wins (§5.7.1). Both MLP and RNN buzzers win against the static threshold baseline, but there is a considerable

Error Reason	Start Count	End Count
Wrong Country	11	17
Wrong Person	16	2
Wrong Place	1	5
Wrong Type	15	5
Wrong Event	0	1
Nonsense	7	2
Annotation	1	4

Table 5.5: The table is an error breakdown for questions with at least twenty-five training examples. To analyze errors at the start of questions, we randomly sampled fifty errors and for end of question took all thirty-six errors. End of question errors are primarily wrong country errors as in Figure 5.16 where the model answers United States instead of Spain. Errors at the start of the question though are more diverse. The most common error is guessing the correct answer type, but not the specific member of that type; examples of this error class include answering Albert Einstein instead of Alan Turing, or Iowa instead of Idaho.

Model	ACC	EW	Score
Threshold		0.013	-9.98
MLP	0.840	0.272	-2.31
RNN	0.849	0.302	-1.01
Optimal	1.0	0.502	2.19

Table 5.6: The accuracy (ACC), expected wins (EW), and QB score (Score) of each buzzer on the validation set. Both MLP and RNN outperform the static threshold baseline by a large margin, but there is still a considerable gap from the optimal buzzer.

gap between RNN and the optimal buzzer.

Low expected wins means the buzzer is either too aggressive or not aggressive enough. To characterize their weaknesses, we compare the buzzers’ behavior over time (Figure 5.18). The static threshold buzzer is too aggressive, especially early in the questions as is also seen in Figure 5.17. This behavior to some extent resonates with the observation that the confidence of neural models needs calibration (Guo et al., 2017). The difference between MLP and RNN is small but RNN is less likely to be overly aggressive early in the question.

For a more fine-grained analysis, we simulate games where our system plays against individual human players using the gameplay dataset (§5.3.3). Based on the guesser, we classify questions as “possible” or not. If the guesser gets the answer correct before the opponent answers, it is *possible* for the buzzer to win the question. Otherwise, it is impossible for the buzzer to do anything to beat the opponent. Based on this categorization, Figure 5.19 further breaks down the outcomes: the RNN is

Test Question: This country seized four vessels owned by Captain John Meares, which were registered in Macau and disguised with Portuguese flags, starting a dispute over fishing rights. To further negotiations with this country, Thomas Jefferson signed the so-called “Two Million Dollar Act.” This country agreed not to police a disputed spot of land, which was subsequently settled by outlaws and “Redbones”, and which was called the “Neutral Ground.” This country was humiliated by England in the Nootka Crisis. Harman Blennerhassett’s farm on an island in the Ohio River was intended as the launching point of an expedition against this European country’s possessions in a plan exposed by James Wilkinson. This country settled navigation rights with the United States in Pinckney’s Treaty, which dealt with the disputed “West” section of a colony it owned. For 10 points, name this European country which agreed to the Adams-Onis Treaty handing over Florida.

Guess: United States **Answer:** Spain

Figure 5.16: Although the answer to this question is Spain, many of the terms and phrases mentioned are correlated with the United States. Thus, the RNN model answers United States instead of the correct answer Spain. This is one of many examples where the model answers with the correct answer type (country), but incorrect member of that type.

less likely to be overly aggressive in both possible and impossible cases.

5.8 Live Exhibition Events

No amount of thorough experimentation and analysis of machine learning systems can match the public interest and rubber-meets-the-road practicalities of live matches between humans and machines. IBM’s Watson in Jeopardy! (Ferrucci et al., 2010), Deep Blue in chess (hsiong Hsu et al., 1995), and Google’s AlphaGo in Go (Silver et al., 2016) were both tremendous scientific achievements and cultural watersheds. In the case of chess and Go, they transformed how the games are played through insight gained from the collaboration of humans and machines. Lastly, although our offline evaluation is reasonable, a live evaluation verifies that the two correspond since that is not always clear (Hersh et al., 2000).

In a similar spirit, we have hosted eight live events since 2015 where we showcase our research to the public by having humans and machines compete against each other.³⁴ Except for our NIPS 2015 Best Demonstration against ML researchers, our system’s opponents have been strong trivia players. Their achievements include victories in numerous national QB championships (high school and college), Jeopardy!, and similar trivia competitions.

Our inaugural event in 2015 at the QB High School National Competition Tournament (HSNCT) pitted an early and vastly different version of our system

³⁴Videos of our events are available at <http://events.qanta.org>.

This instrument plays the ▲ only extended solo in the overture to Verdi's ▼¹ Luisa Miller. This is the solo instrument in a piece ■ that opens with the movement “The Perilous Shore”. ◆ This instrument introduces ● the main theme to “The Pines of Janiculum” from Respighi’s The Pines of Rome. This instrument has a long solo at the beginning of the Adagio from Rachmaninoff’s Second Symphony, and it first states the Shaker theme in Copland’s Appalachian Spring. John Adams’ Gnarly Buttons is for this instrument. Heinrich ▼ Baermann, a virtuoso on this instrument, was the ▼ dedicatee of Carl Maria von Weber’s two concertos for it. The basset ▼ horn is a variant of, for 10 points, what ▼² ▼ single-reed woodwind instrument, which plays ▼ a notable glissando at the opening of Gershwin’s ▼³ Rhapsody in Blue?

Answer: Clarinet **Optimal Buzz:** ◆ **Correct:** ▼▲●■ **Wrong:** ▼▲●■

Threshold Buzz: ▲ **Bassoon MLP Buzz:** ■ **Bassoon RNN Buzz:** ●

Clarinet

Human Buzzes: ▼¹ Violin, ▼² Obo, ▼³ Flute, ▼ Clarinet

Figure 5.17: In this question, the Threshold ▲ and MLP ■ buzzers are too aggressive and buzz before the guesser’s answer is correct. In contrast, the RNN ● is more conservative and buzzes shortly after the optimal point ◆ which is—by a wide margin—still earlier than the earliest (correct) human buzz ▼.

against a team of tournament organizers in a match that ended in a tie.³⁵ Later that year, a similar system defeated Ken Jennings of Jeopardy! fame at the University of Washington, but lost convincingly (145–345) at HSNCT 2016. The subsequent year at HSNCT 2017, our redesigned system narrowly defeated its opponents (260–215). This system used an IR guesser combined with a question type classifier (Li and Roth, 2002) and an RNN buzzer.³⁶ Although this impressive result appears to follow the trend of machines improving until they defeat skilled humans, it is far from the whole story as we will see in Chapter 6.

In parallel with these events, we hosted events where teams—humans and machines—were selected from open competition. Our first of these style events was hosted as part of a NAACL 2016 workshop on question answering. Before the event, local high school teams competed against each other, and researchers submitted their machine systems which also played simulated matches against each other. At the event the best human and machine teams played against each other with the high school team defeating an early version of Studio OUSIA’s system (Yamada et al., 2017, 2018b).³⁷ In 2017, we hosted a similar workshop at NIPS where an improved

³⁵Our software did not handle ties correctly and terminated instead of playing tiebreaker questions.

³⁶In absolute terms, the type classifier did not improve accuracy; however, in our matches we display the top five scoring guesses, and the type classifier improved the “plausibility” of that list.

³⁷The OUSIA system embeds words and entities separately, and uses a DAN-based architecture over these.

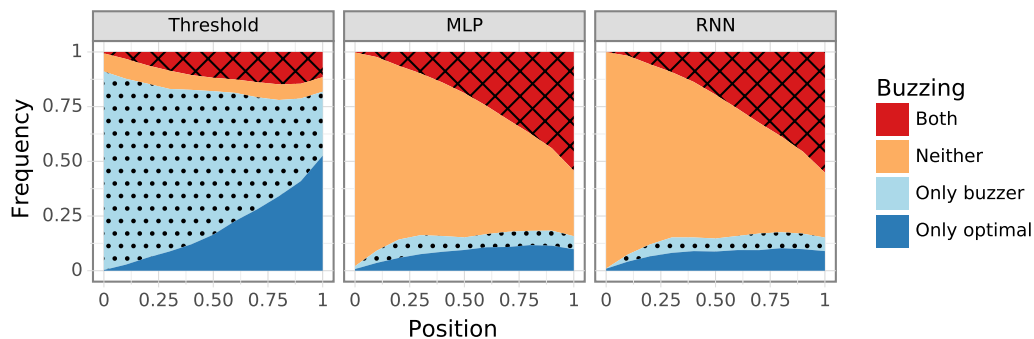


Figure 5.18: Comparing buzzers’ behavior over time against the optimal buzzer. The red crossed area and dotted blue area combined indicates when the buzzer thinks that the guesser is correct, the other two combined when the buzzer thinks the guesser is wrong. The red (crossed) and orange (unhatched) areas combined indicates when the buzzer matches the optimal buzzer. Our goal is to maximize the red areas and minimize the blue areas. The static threshold baseline is overly aggressive, especially at earlier positions in the question (large dotted blue area); MLP and RNN both behaves reasonably well, and the aggressiveness of RNN is slightly more balanced early on in the question.

version of OUSIA’s system yet again defeated its machine competition, but this time also defeated the invited human team.

Events and collaborations like these show that QB is more than just another question answering task. By engaging with the QB community to digitize QB questions in a machine-readable form we not only made our research possible, but started the ecosystem of tools that students now rely on to practice with before competitions. In the next step towards deeper collaboration with this community we are building ways for humans and machines to cooperate in competition (Feng and Boyd-Graber, 2019) and in writing questions (Chapter 6). We accomplish all this while simultaneously providing ways for students of all ages to engage with and benefit from research through our live exhibition events.

5.9 Related Work

Quizbowl is a question answering and sequential decision-making task. Where Chapter 2 provides a general review of QA tasks, this section compares and contrasts QB to some of these tasks and datasets. Following this, we make connections between the decision-making aspect of QB to model calibration and prediction under domain shift. Having discussed the uncertainty in model confidence scores, we finally discuss handling uncertainty caused by opponents through opponent modeling.

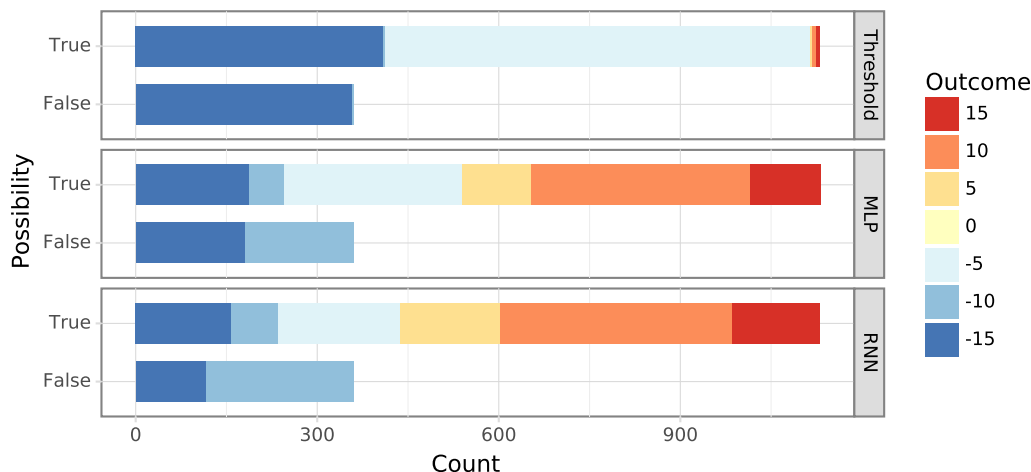


Figure 5.19: Breaking down the buzzer’s performance on the individual question level. Impossible question means there is nothing the buzzer can do to beat the opponent. It is clearer that RNN performs better than MLP, making fewer mistakes of being overly aggressive.

5.9.1 Question Answering Datasets

In our comparison, we focus on *factoid* QA tasks in particular. The least complex types of questions are often called “simple questions” since they can be answered by a single simple fact. For example, SimpleQuestions (Bordes et al., 2015) is specifically designed so that questions can be answered using one knowledge-base triplet, and WikiMovies (Miller et al., 2016b) automatically generates questions from knowledge base triplets. Similarly, WebQuestions (Berant et al., 2013) uses the Google Suggest API to collect questions containing specific entities and crowdworkers only answer questions using Freebase facts. Despite the relative ease in creating these datasets, they lack the complexity and linguistic diversity of “natural” data which is partly responsible for tasks like SimpleQuestions being essentially solved (Petrochuk and Zettlemoyer, 2018). In QB, the capability to answer simple question like these is a *bare minimum requirement* and takes form as the final, “giveaway” clues at the end of questions which even novice (human) players usually answer correctly.

Humans have played trivia games and tournaments for decades (Boyd-Graber and Börschinger, 2020) and as a result there are ample non-QB sources of trivia questions. The most famous example—Jeopardy!—was converted into the SearchQA dataset (Dunn et al., 2017). Other trivia-based QA datasets include TriviaQA which is built from fourteen trivia sites (Joshi et al., 2017) and Quasar-T which is built from questions collected by a reddit user (Dhingra et al., 2017). While some of these are paired with potentially useful supporting evidence, a hallmark of these datasets and QB is that the question alone unambiguously identifies an answer.

However, providing supporting documents to answer questions is another popular way to frame QA tasks. Although TrecQA (Voorhees and Tice, 2000) and Natu-

ralQuestions (Kwiatkowski et al., 2019) differ in that answers are not known a priori by the asker (§2.2.3), they are good examples of questions that are paired with verified supporting documents. In these tasks, the questions—user queries from search engines—were written without knowledge of any particular supporting documents and afterwards annotators attempted to find appropriate supporting documents. Although TriviaQA provides potentially relevant documents, they are not human verified; similarly, MS MARCO (Nguyen et al., 2016), WikiReading (Hewlett et al., 2016), and NewsQA (Trischler et al., 2017) also provide unverified supporting documents. SQuAD in particular falls outside this paradigm since—in general—questions are dependent on the selected context paragraph (§2.2.2).

Another set of tasks focuses on creating questions that require multiple supporting documents and multi-step reasoning. In QB, early clues are often a composition of multiple facts about the answer. For example, to guess “Die Zauberflöte” from “At its premiere, the librettist of this opera portrayed a character who asks for a glass of wine with his dying wish”, one would have to combine two pieces of text from Wikipedia: “Papageno enters. The priests grant his request for a glass of wine and he expresses his desire for a wife.” and “Emanuel Schikaneder, librettist of Die Zauberflöte, shown performing in the role of Papageno”. While this work focuses on tossup QB questions, a second type of QB question—bonuses—emphasize this multi-step aspect through multi-part questions (Elgohary et al., 2018).

Multi-step reasoning datasets primarily differ by providing ground-truth annotations to sufficient supporting documents. In Wikihop (Welbl et al., 2018), multi-hop questions are automatically constructed from Wikipedia, Wikidata, and WikiReading. HotPotQA (Yang et al., 2018b) follows a similar structure, but question text is crowdsourced rather than automatically generated. However, Min et al. (2019) show that although these questions are meant to be only solvable by using multi-step reasoning, that many are answerable with single-hop reasoning. Subsequent datasets like QASC (Khot et al., 2020), DROP (Dua et al., 2019), and BREAK (Wolfson et al., 2020) focus on creating questions that are much more likely to require multi-step reasoning through adversarial annotation. However, multi-step questions are not the only way to make questions more difficult.

Adversarial question authoring and adversarial filtering (Zellers et al., 2018) are other ways to increase difficulty. The general framework of adversarial authoring filters questions either during or after annotation by whether a strong baseline answers it correctly. For example, Wallace et al. (2019b) show that when QB question writers—while authoring a question—are shown what a model would answer and why, that they create questions that are significantly more difficult for machines while being no more difficult for human players. Along similar lines, Bartolo et al. (2020) let writers see what a model would answer, but iteratively create new adversarial questions, re-train the model, and collect new adversarial questions anew. Although contrast sets explicitly do not have a model in the loop (Gardner et al., 2020a), they are similarly intended to challenge models through example perturbations. Other effective example perturbations for automatically creating adversarial examples include adding sentences to context paragraphs (Jia and Liang, 2017) and in general semantic-preserving permutations (Ribeiro et al., 2018). The common

thread in all these works is to make questions more difficult for machines so that models continue to improve.

5.9.2 Answer Triggering and Model Calibration

Just as in QB’s buzzer, in real-world applications of machine learning it is important to know when to trust a model’s confidence in its predictions (Jiang et al., 2012). In QA, knowing when to answer is known as answer triggering (Yang et al., 2015; Rajpurkar et al., 2018) and the core task—correctly estimating model confidence—is model calibration (Zadrozny and Elkan). Having machines that accurately convey their confidence is doubly important since humans have the unfortunate tendency to place too much trust in machines (Sundar, 2007). Despite this, the rise of deep learning seems to have only made this more challenging (Guo et al., 2017; Feng et al., 2018). Fortunately, recent work has made progress in this problem by making connections to out-of-domain detection (Kamath et al., 2020). Along similar lines, the buzzing task in QB is partially dependent on having well-calibrated models.

5.9.3 Opponent Modeling

QB is far from the only game where players benefit from modeling opponent behavior. Opponent modeling is particularly important in games with hidden information—in QB this takes the form of the yet-to-be-revealed question and the per-question skill of the opponent. One classic example of opponent modeling under uncertainty is Poker (Billings et al., 1998) where identifying and adapting to opponents is central to the game. In games like Scrabble, opponent behavior can even be used to infer hidden information—such as their remaining tiles—to make it easier to anticipate and counter their strategy (Richards and Amir, 2007). Similarly, in real-time strategy games the opponent’s strategy is often hidden and hierarchically structured (Schadd et al., 2007); they may play aggressive—similar to aggressive buzzing—making defensive play more advantageous. Traditionally, QB is played in teams, so a full model should account for your own team’s skills as well as the mixture of opponent skills (e.g., certain players may be better at history questions). Although this is unaddressed in QB, this general framing has been considered in games where players compete with each other for limited resources, but are advantaged by doing so semi-cooperatively (Von Der Osten et al., 2017). One potential area of future work is in focusing on opponent modeling in QB with an emphasis on accounting for teams of players and strategies that evolve over the course of a match (e.g., play less conservatively when trailing).

5.10 Future Work

Since future work in Chapter 7 focuses on directions to improve dataset construction and QA evaluations, here we identify research directions that are particularly well suited to QA in particular. Improving QB models can incorporate many

commonplace tasks in NLP other than question answering. Reasoning about entities can be improved through better named entity recognition, entity linking, coreference and anaphora resolution. Some of the more difficult clues in QB however presume that the player has read and integrated the content of books such as important plot points. Further work in reading comprehension and summarization could help answer some of these questions. At a more general level, the extraction of information from external knowledge sources (such as books or Wikipedia) is important since the distribution of training examples per answer is heavily skewed, and some new questions ask about current events. Next, we take a closer look at future work using the perpetual influx of new and diverse questions from annual tournaments and working with the supportive QB community.

5.10.1 Generalization in Factoid Question Answering

Although the precise format of QA is a bit idiosyncratic, it continues to grow in size and diversity year-over-year due to continually running tournaments and the factual arms race between players and writers (§5.2.3). With the popularization of QB, the number of questions per year quadrupled in the ten-year period between 2007 and 2017 (Figure 5.20). As the dataset continues to grow, it will demand that machines and humans broaden their knowledge of past events while also updating their knowledge with current events. Every year presents an opportunity to test both how well models generalize to novel questions and how well they generalize to questions about current events. For example, in a 2017 exhibition match our model missed a question about the company responsible for driving down rocket launch costs (SpaceX); a phenomenon which only manifested itself several years prior. These questions could become part of the effort to build dynamic leaderboards like in Chapter 7 which will hopefully improve the generalization of QA models.

For QB specifically, a major unsolved challenge is few-shot and zero-shot QA. The scarcity of training examples per unique answer (Figure 5.14) causes a substantial drop in the accuracy of QB models (Figure 5.15). The central challenge is one that every student agonizes over while reviewing for exams: anticipating what topics will be on the exam! In QB, “being able to predict and learn about the non-canonical answers that are on the cusp of being asked about is one way to become a great player”(can, 2020). In our data and throughout exhibition matches, we estimate that between 15-20% of questions have novel answers. A similar challenge is faced in computer vision where not all object categories are known beforehand and for many there is scarce training data (Xian et al., 2018). Although there are relatively simple domain adaptation methods based on feature augmentation (Daume III, 2007; Kim et al., 2016b) and adversarial learning (Chen and Cardie, 2018), these still do not address the core challenge of anticipating test-time topics. Recent work provides a potential solution to this by constructing a large set of “probably-asked questions” (Lewis et al., 2021) to use for training. Beyond this, although there is a cultural understanding of the “canon” of QB, this has not been rigorously studied, which could serve to inform how to best train zero-shot QA models for QB, but more interestingly shed insight into how a dedicated community’s shared memory

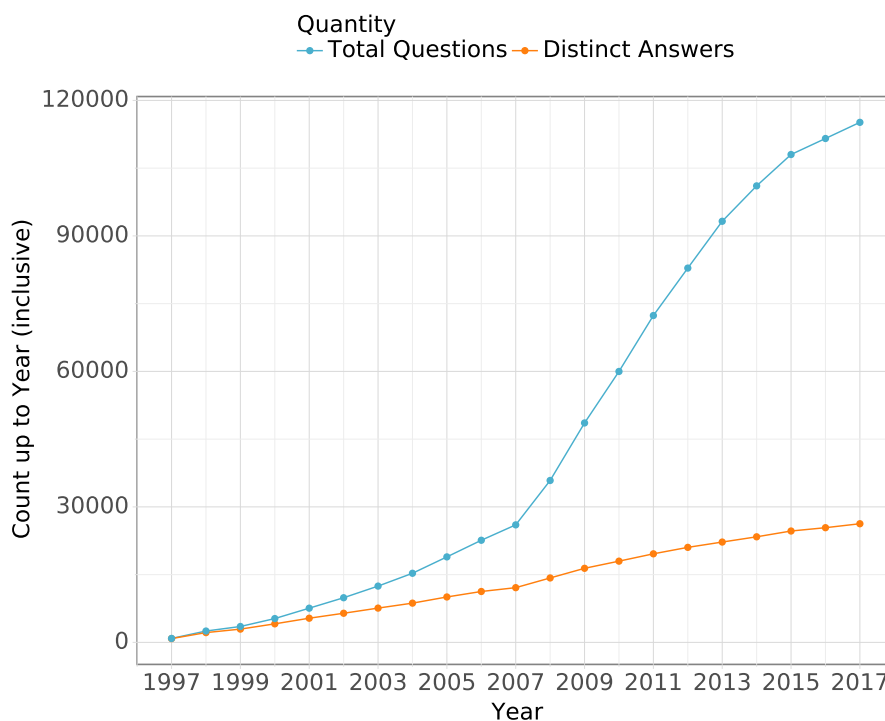


Figure 5.20: The growth of the QANTA dataset in number of questions and number of distinct answers over the past twenty years starting in 1997. The dataset has grown by at least 5,000 questions every year since 2010. All questions with matched answers are included, and we construct the plot by using the tournament year of each question. Independently, participation in QB (and thus number of students writing questions) has roughly doubled every year since 2008.

has evolved over time.

5.10.2 Robust, Trustable, and Explainable Machine Learning

QB naturally encourages QA systems that are well calibrated, but is a useful setting for improving robustness (Goel et al., 2021), explainability (Belinkov and Glass, 2019) and trust (Mitchell et al., 2019) in machine learning systems. For example, QA is also a good platform evaluating interpretations through machine-human cooperative play (Feng and Boyd-Graber, 2019). Beyond this though, as Chapter 6 explores, QB is a good platform for testing the explainability of QA systems through adversarial authoring. Our collaborative research in QB thus far is only a beginning.

5.11 Conclusion

This chapter introduces and makes the case for incremental, pyramidal question answering with QB. Solving QB questions requires sophisticated NLP such as

resolving complex coreference, multi-hop reasoning, and understanding the relationships between the plethora of entities that could be answers. Fundamental to answering QB questions is that the questions are incremental; this is both fun and good for research. It is fun because it allows for live, engaging competitions between humans and computers. To evaluate systems we use three methods: offline accuracy-based metrics adapted to the incremental nature of QB, simulated matches against machines and humans, and live exhibition matches. Although the best models have sixty percent accuracy at the end of questions, this is well below the best players, and early clues remain particularly challenging. This format—the product of refining human question answering competitions over decades—is also good for research because it allows for fair, comprehensive comparison of systems and iterative improvement as systems answer questions earlier and earlier.

However, the benefits to research go beyond format or specific sub-tasks, and extend to our symbiotic collaboration with the public. Exhibition matches double as outreach events and opportunities to put machine systems to the test on previously unseen questions. In this next chapter, we show that by collaborating with the QB community we can combine the strengths of machines and humans to improve question quality. Rather than compete against QB systems, writers collaborate with machine learning tools to discover bad clues which helps create questions more interesting to humans that also better test the generalization of systems. By aligning the goals of the trivia and QA communities, we can create datasets that better discriminate different levels of language and knowledge understanding.

Beyond the specific QB community, live exhibitions also serve the general public. Exhibition games demonstrate what NLP and ML systems can do and what they cannot. When the tricks that computers use to answer questions are revealed to lay audiences, some of the mystique is lost, but it can also encourage enthusiasts to investigate our techniques to see if they can do better. Our open data and code facilitates this open competition, and the QB community helps make it fun and engaging.

QB isn't just another dataset or task; it is a rich platform for NLP research that co-evolves with the QB community. One example of this is our digitization of QB questions through our online interface which created and popularized online QB play. The next chapter explores how machines and humans can collaborate to write better questions while advancing NLP research. Beyond this, we hope that new unforeseen research directions will continue emerging from this collaboration while simultaneously giving back to the QB community through new and exciting ways of engaging with state-of-the-art research in machine learning and natural language processing.

Chapter 6: Centaur Authoring of Adversarial Questions

Similar to how the mythological centaur was half-human, half-horse, [these chess teams] were half-human, half-AI. Because humans & AIs are strong on different dimensions, together, as a centaur, they can beat out solo humans and computers alike.

Nicky Case on Kasparov’s Centaur Chess

This chapter continues building on the idea of creating more discriminative questions, but instead of focusing on format as in Chapter 5, we focus on more discriminative data.¹ As Chapter 4 points out (§4.4.5), questions are least discriminative when the likelihood of both subjects having the same response is highest because the question is either too easy or too hard. One thing that can make QB questions too easy for machines is if statistical patterns give away the answer (§5.7.2); the end effect is that an otherwise challenging question to make progress on is wasted. This chapter introduces an annotation framework where humans and machines cooperate to write (QB) questions that contain fewer of these statistical patterns. Although this may be undesirable under the Cranfield paradigm (§2.1.1),² under the Manchester paradigm (§2.1.2) this is important since under that paradigm the presence (or absence) of specific statistical patterns should not matter.

6.1 Introduction

Proponents of machine learning claim human parity on tasks like reading comprehension (Yu et al., 2018) and commonsense inference (Devlin et al., 2018). Despite these successes, many evaluations neglect that computers solve natural language processing (NLP) tasks in a fundamentally different way than humans.

Models can succeed without developing “true” language understanding (Bender and Koller, 2020), instead learning superficial patterns from crawled (Chen et al., 2016) or manually annotated datasets (Kaushik and Lipton, 2018; Gururangan et al., 2018). In the context of testing for intelligent behavior, answering from patterns alone is undesirable. Thus, recent work stress tests models via adversarial evaluation: elucidating a system’s capabilities by exploiting its weaknesses (Jia and Liang, 2017; Belinkov and Glass, 2019). Unfortunately, while adversarial evaluation reveals

¹This chapter is based on the TACL publication Wallace et al. (2019b).

²Some patterns may be useful; at the same time, IR systems should be robust so the desirability of statistical patterns is contextual.



Figure 6.1: Adversarial evaluation in NLP typically focuses on a specific phenomenon (e.g., word replacements) and then generates the corresponding examples (top). Consequently, adversarial examples are limited to the diversity of what the underlying generative model or perturbation rule can produce and also require downstream human evaluation to ensure validity. Our setup (bottom) instead has human-authored examples, using human–computer collaboration to craft adversarial examples with greater diversity.

simplistic model failures (Ribeiro et al., 2018; Mudrakarta et al., 2018), exploring more complex failure patterns requires human involvement (Figure 6.1): automatically modifying natural language examples without invalidating them is difficult. Hence, the diversity of adversarial examples is often severely restricted.

Instead, our half-human, half-computer (centaur) approach uses human creativity to generate adversarial examples. A user interface presents model interpretations and helps users craft model-breaking examples (§6.3). We apply this to a QB (Chapter 5) where trivia enthusiasts—who *already* write questions for academic competitions—create diverse examples that stump existing QA models.

The adversarially authored test set is nonetheless as easy as regular questions for humans (§6.4), but the relative accuracy of strong QA models drops as much as 40% (§6.5). We also host live human vs. computer matches—where models typically defeat top human teams—but unlike our prior exhibition matches (§5.8) observe spectacular model failures on adversarial questions.

Analyzing the adversarial edits uncovers phenomena that humans can solve but computers cannot (§6.6), validating that our framework uncovers creative, targeted adversarial edits (§6.7). Our resulting adversarial dataset presents a fun, challenging, and diverse resource for future QA research: a system that masters it will demonstrate more robust language understanding.

6.2 Adversarial Evaluation for NLP

Adversarial examples (Szegedy et al., 2013) often reveal model failures better than traditional test sets. However, automatic adversarial generation is tricky for NLP (e.g., by replacing words) without changing an example’s meaning or invalidating it.

Recent work side-steps this by focusing on simple transformations that preserve meaning. For instance, Ribeiro et al. (2018) generate adversarial perturbations such as replacing *What has* → *What’s*. Other minor perturbations such as typos (Belinkov and Bisk, 2018), adding distractor sentences (Jia and Liang, 2017; Mudrakarta et al., 2018), or character replacements (Ebrahimi et al., 2018) preserve meaning while degrading model performance.

Generative models can discover more adversarial perturbations but require *post hoc* human verification of the examples. For example, neural paraphrase or language models can generate syntax modifications (Iyyer et al., 2018), plausible captions (Zellers et al., 2018), or NLI premises (Zhao et al., 2018). These methods improve example-level diversity but mainly target a specific phenomenon, e.g., rewriting question syntax.

Furthermore, existing adversarial perturbations are restricted to sentences—not the paragraph inputs of QB and other tasks—due to challenges in long-text generation. For instance, syntax paraphrase networks (Iyyer et al., 2018) applied to QB only yield valid paraphrases 3% of the time (Appendix D.1).

6.2.1 Putting a Human in the Loop

Instead, we task human authors with *adversarial writing* of questions: generating examples which break a specific QA system but are still answerable by humans. We expose model predictions and interpretations to question authors, who find question edits that confuse the model.

The user interface makes the adversarial writing process interactive and model-driven, in contrast to adversarial examples written independent of a model (Ettinger et al., 2017). The result is an adversarially authored dataset that explicitly exposes a model’s limitations by design.

Human-in-the-loop generation can replace or aid model-based adversarial generation approaches. Creating interfaces and interpretations is often easier than designing and training generative models for specific domains. In domains where adversarial generation is feasible, human creativity can reveal which tactics automatic approaches can later emulate. Model-based and human-in-the-loop generation approaches can also be combined by training models to mimic human adversarial edit history, using the relative merits of both approaches.

6.3 Our QA Testbed: Quizbowl

Like the work in Chapter 5, we continue using QB as it is the “gold standard” of academic competitions between universities and high schools is QB.

6.3.1 Known Exploits of Quizbowl Questions

Like most QA datasets, QB questions are written for *humans*. Unfortunately, the heuristics that question authors use to select clues do not always apply to computers. For example, humans are unlikely to memorize every song in every opera by a particular composer. This, however, is trivial for a computer. In particular, a simple QA system easily solves the example in Figure 6.2 from seeing the reference to “Un Bel Di”. Other questions contain uniquely identifying “trigger words” (Harris, 2006). For example, “martensite” only appears in questions on steel. For these examples, a QA system needs only use a simple and non-generalizable if-then rule.

The protagonist of this opera describes the future day when her lover will arrive on a boat in the aria “Un Bel Di” or “One Beautiful Day”. The only baritone role in this opera is the consul Sharpless who reads letters for the protagonist, who has a maid named Suzuki. That protagonist blindfolds her child Sorrow before stabbing herself when her lover B. F. Pinkerton returns with a wife. For 10 points, name this Giacomo Puccini opera about an American lieutenant’s affair with the Japanese woman Cio-Cio San.

Answer: Madama Butterfly

Figure 6.2: This question is easily answered by a model after seeing the reference to “Un Bel Di.” Our adversarial writing process highlights terms like these which humans can then modify to make clues more challenging for computer.

One might wonder if this means that factoid QA is thus an uninteresting, nearly solved research problem. However, some QB questions are fiendishly difficult for computers. Many questions have intricate coreference patterns (Guha et al., 2015), require reasoning across multiple types of knowledge, or involve complex wordplay. If we can isolate and generate questions with these difficult phenomena, “simplistic” factoid QA quickly becomes non-trivial.

6.3.2 Models and Datasets

We conduct two rounds of adversarial writing. In the first, authors attack the traditional information retrieval (IR) system from Section 5.5.1 which we also used in the NIPS 2017 QB shared task (Boyd-Graber et al., 2018).

In the second round, authors attack either the IR model or a neural QA model. The neural model is a bidirectional RNN using the gated recurrent unit architecture (Cho et al., 2014b) (§5.5.3). Both models in this round are trained using the full QANTA dataset (§5.3). The second round dataset incorporates more diverse answers than the first (25,000 entities versus 11,000 in round one).³

6.3.3 Interpreting Quizbowl Models

To help write adversarial questions, we expose what the model is thinking to the authors. We interpret models using saliency heat maps: each word of the question is highlighted based on its importance to the model’s prediction (Ribeiro et al., 2016).

For the neural model, word importance is the decrease in prediction probability when a word is removed (Li et al., 2016; Wallace et al., 2018). We focus on gradient-based approximations (Simonyan et al., 2014; Montavon et al., 2018) for their computational efficiency. To interpret a model prediction on an input sequence of n words $\mathbf{w} = \langle \mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_n \rangle$, we approximate the classifier f with a linear function of w_i derived from the first-order Taylor expansion. The importance of w_i , with

³The QANTA dataset was built contemporaneously with the initial data collection of this chapter, and we used the improved version in the second data collection round.

The screenshot displays the Madama Butterfly QA interface. At the top right, the question is written: "The protagonist of this opera describes the future day when her lover will arrive on a boat in the aria "Un Bel Di" or "One Beautiful Day." The only baritone role in this opera is the consul Sharpless who reads letters for the protagonist, who has a maid named Suzuki. That protagonist blindfolds her child Sorrow before stabbing herself when her lover B.F. Pinkerton returns with a wife. For 10 points, name this Giacomo Puccini opera about an American lieutenant's affair with the Japanese woman Cio-Cio San." A "Submit" button is visible. On the top left, the "Machine Guesses" section shows a table of the top five model predictions:

#	Guess	Confidence
1	Madama Butterfly	0.74
2	Giacomo Puccini	0.03
3	Andrea Chénier	0.02
4	La traviata	0.02
5	NoRMA	0.02

Below the guesses is a "Settings" panel with options like "Don't release questions" and "Provide Automatic Updates Every 5 Words". At the bottom right, the "Evidence for Madama Butterfly" section shows a table of evidence snippets with highlighted words from the question and the evidence text.

Figure 6.3: The author writes a question (top right), the QA system provides guesses (left), and explains why it makes those guesses (bottom right). The author can then adapt their question to “trick” the model.

embedding \mathbf{v}_i , is the derivative of f with respect to the one-hot vector:

$$\frac{\partial f}{\partial w_i} = \frac{\partial f}{\partial \mathbf{v}_i} \frac{\partial \mathbf{v}_i}{\partial w_i} = \frac{\partial f}{\partial \mathbf{v}_i} \cdot \mathbf{v}_i. \quad (6.1)$$

This simulates how model predictions change when a particular word’s embedding is set to the zero vector—it approximates word removal (Ebrahimi et al., 2018; Wallace et al., 2018).

For the IR model, we use the ElasticSearch Highlight API (Gormley and Tong, 2015), which provides word importance scores based on query matches from the inverted index.

6.3.4 Adversarial Writing Interface

The authors interact with either the IR or RNN model through a user interface⁴ (Figure 6.3). An author writes their question in the upper right while the model’s top five predictions (*Machine Guesses*) appear in the upper left. If the top prediction is the right answer, the interface indicates where in the question the model is first correct. The goal is to cause the model to be incorrect or to delay the correct answer position as much as possible.⁵ The words of the current question are highlighted using the applicable interpretation method in the lower right (*Evidence*). We do not

⁴<https://github.com/Eric-Wallace/trickme-interface/>

⁵The authors want normal QB questions which humans can easily answer by the very end. For popular answers, (e.g., Australia or Suez Canal), writing novel final give-away clues is difficult. We thus expect models to often answer correctly by the very end of the question.

enforce time restrictions or require questions to be adversarial: if the author fails to break the system, they are free to “give up” and submit any question.

The interface continually updates as the author writes. We track the question edit history to identify recurring model failures (§6.6) and understand how interpretations guide the authors (§6.7).

6.3.5 Question Authors

We focus on members of the QB community: they have deep trivia knowledge and craft questions for QB tournaments (§5.2.3). We award prizes for questions read at live human–computer matches (§6.5.3).

The question authors are familiar with the standard format of QB questions (Lujan and Teitler, 2003). The questions follow a common paragraph structure, are well edited for grammar, and finish with a simple “give-away” clue (§5.2.2). These constraints benefit the adversarial writing process as it is very clear what constitutes a difficult but valid question. Thus, our examples go beyond surface-level “breaks” such as character noise (Belinkov and Bisk, 2018) or syntax changes (Iyyer et al., 2018). Rather, questions are difficult because of their semantic content (examples in Section 6.6).

6.3.6 How an Author Writes a Question

To see how an author might write a question with the interface, we walk through an example of writing a question’s first sentence. The author first selects the answer to their question from the training set—Johannes Brahms—and begins:

Karl Ferdinand Pohl showed this composer some pieces on which this composer’s Variations on a Theme by Haydn were based.

The QA system *buzzes* (i.e., it has enough information to interrupt and answer correctly) after “composer”. The author sees that the name “Karl Ferdinand Pohl” appears in Brahms’ Wikipedia page and avoids that specific phrase, describing Pohl’s position instead of naming him directly:

This composer was given a theme called “Chorale St. Antoni” by the archivist of the Vienna Musikverein, which could have been written by Ignaz Pleyel.

This rewrite adds in some additional information (there is a scholarly disagreement over who wrote the theme and its name), and the QA system now incorrectly thinks the answer is Frédéric Chopin. The user can continue to build on the theme, writing

While summering in Tutzing, this composer turned that theme into “Variations on a Theme by Haydn”.

Science	17%
History	22%
Literature	18%
Fine Arts	15%
Religion, Mythology, Philosophy, and Social Science	13%
Current Events, Geography, and General Knowledge	15%
Total Questions	1213

Table 6.1: The topical diversity of the questions in the adversarially authored dataset based on a random sample of 100 questions.

Again, the author sees that the system buzzes “Variations on a Theme” with the correct answer. However, the author can rewrite it in its original German, “Variationen über ein Thema von Haydn” to fool the system. The author continues to create entire questions the model cannot solve.

6.4 A New Adversarially-Authored Dataset

Our adversarial dataset consists of 1213 questions with 6,541 sentences across diverse topics (Table 6.1).⁶ There are 807 questions written against the IR system and 406 against the neural model by 115 unique authors. We plan to hold twice-yearly competitions to continue data collection.

6.4.1 Validating Questions with Quizbowlers

We validate that the adversarially authored questions are not of poor quality or too difficult for humans. We first automatically filter out questions based on length, the presence of vulgar statements, or repeated submissions (including re-submissions from the QB training or evaluation data).

We next host a human-only QB event using intermediate and expert players (former and current collegiate QB players). We select sixty adversarially authored questions and sixty standard high school national championship questions, both with the same number of questions per category (list of categories in Table 6.1).

To answer a QB question, a player interrupts the question: the earlier the better. To capture this dynamic, we record both the average answer position (as a percentage of the question, lower is better) and answer accuracy. We shuffle the regular and adversarially authored questions, read them to players, and record these two metrics.

The adversarially authored questions are on average *easier* for humans than the regular test questions. For the adversarially authored set, humans buzz with

⁶Data available at <http://trickme.qanta.org>.

41.6% of the question remaining and an accuracy of 89.7%. On the standard questions, humans buzz with 28.3% of the question remaining and an accuracy of 84.2%. The difference in accuracy between the two types of questions is not significantly different ($p = 0.16$ using Fisher’s exact test), but the buzzing position is earlier for adversarially authored questions ($p = 0.0047$ for a two-sided t -test). We expect the questions that were not played to be of comparable difficulty because they went through the same submission process and post-processing. We further explore the human-perceived difficulty of the adversarially authored questions in Section 6.5.3.

6.5 Computer Experiments

This section evaluates QA systems on the adversarially authored questions. We test three models: the IR and RNN models shown in the interface, as well as a Deep Averaging Network (§5.5.3) to evaluate the transferability of the adversarial questions. We break our study into two rounds. The first round consists of adversarially authored questions written against the IR system (§6.5.1); the second round questions target both the IR and RNN (§6.5.2).

Finally, we also hold live competitions that pit the state-of-the-art Studio Ousia model (Yamada et al., 2018b) against human teams (§6.5.3).

6.5.1 First Round Attacks: IR Adversarial Questions Transfer To All Models

The first round of adversarially authored questions target the IR model and are significantly harder for the IR, RNN, and DAN models (Figure 6.4). For example, the DAN’s accuracy drops from 54.1% to 32.4% on the full question (60% of original performance).

For both adversarially authored and original test questions, early clues are difficult to answer (accuracy about 10% for the first quarter of the question). However, during the middle third of the questions, where buzzes in QB most frequently occur, accuracy on original test questions rises quicker than the adversarially authored ones. For both, the accuracy rises towards the end as the clues become “give-aways”.

6.5.2 Second Round Attacks: RNN Adversarial Questions are Brittle

In the second round, the authors also attack an RNN model. All models tested in the second round are trained on a larger dataset (§6.3.2).

A similar trend holds for IR adversarial questions in the second round (Figure 6.5): a question that tricks the IR system also fools the two neural models (i.e., adversarial examples transfer). For example, the DAN model was never targeted but had substantial accuracy decreases in both rounds.

However, this does not hold for questions written adversarially against the RNN model. On these questions, the neural models struggle but the IR model is largely unaffected (Figure 6.5, right).

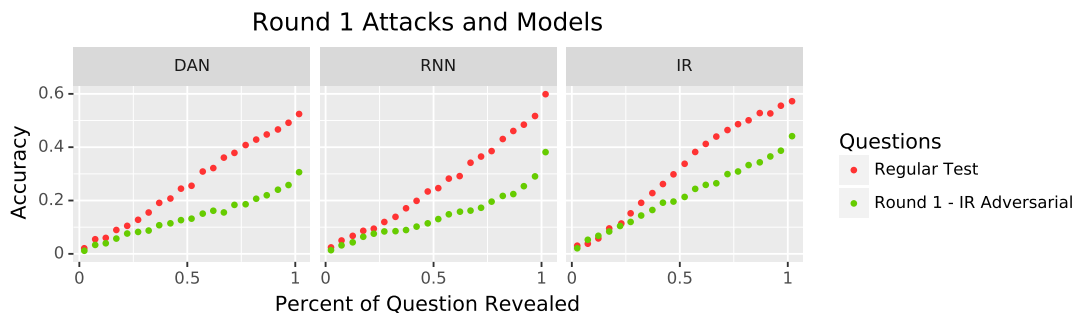


Figure 6.4: The first round of adversarial writing attacks the IR model. Like regular test questions, adversarially authored questions begin with difficult clues that trick the model. However, the adversarial questions are significantly harder during the crucial middle third of the question.

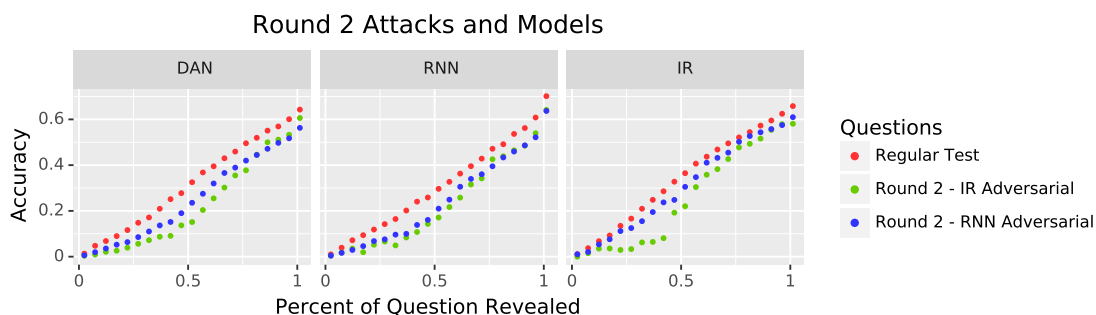


Figure 6.5: The second round of adversarial writing attacks the IR and RNN models. The questions targeted against the IR system degrade the performance of all models. However, the reverse does not hold: the IR model is robust to the questions written to fool the RNN.

6.5.3 Humans vs. Computer, Live (again)!

In the offline setting (i.e., no pressure to “buzz” before an opponent) models demonstrably struggle on the adversarial questions. But, what happens in standard QB: live, head-to-head games?

We run two live humans vs. computer matches. The first match uses IR adversarial questions in a forty question, tossup-only QB format. We pit a human team of national-level QB players against the Studio Ousia model (Yamada et al., 2018b), the current state-of-the-art QB system. The model combines neural, IR, and knowledge graph components (details in Appendix D.2), and won the 2017 NIPS shared task, defeating a team of expert humans 475–200 on regular QB test questions. Although the team at our live event was comparable to the NIPS 2017 team, the tables were turned: the human team won handedly 300–30.

Our second live event is significantly larger: seven human teams play against models on over 400 questions written adversarially against the RNN model. The human teams range in ability from high school QB players to national-level teams (Jeopardy! champions, Academic Competition Federation national champions, top

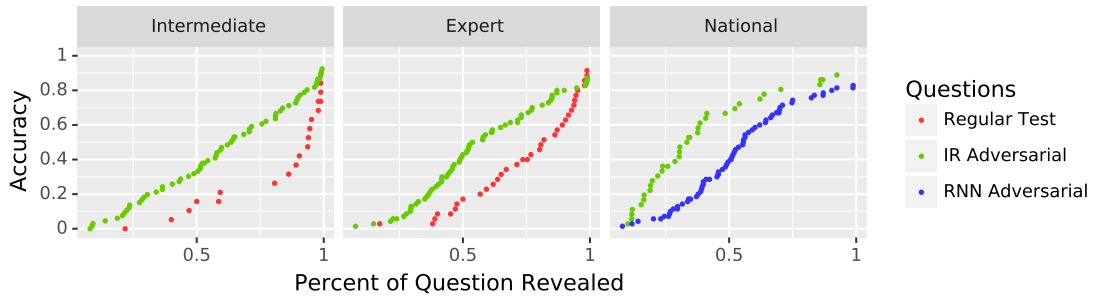


Figure 6.6: Humans find adversarially authored question about as difficult as normal questions: rusty weekend warriors (*Intermediate*), active players (*Expert*), or the best trivia players in the world (*National*).

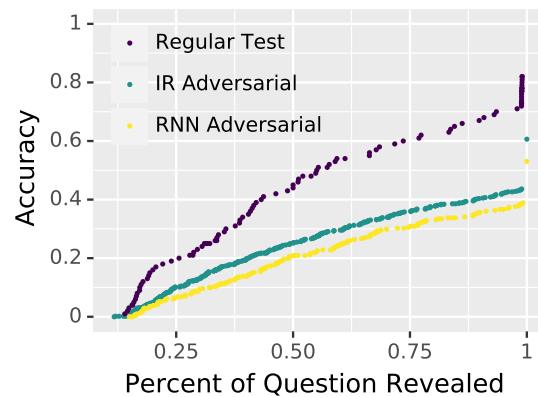


Figure 6.7: The accuracy of the state-of-the-art Studio Ousia model degrades on the adversarially authored questions despite never being directly targeted. This verifies that our findings generalize beyond the RNN and IR models.

scorers in the World Quizzing Championships). The models are based on either IR or neural methods. Despite a few close games between the weaker human teams and the models, humanity prevailed in every match.⁷

Figures 6.6–6.7 summarize the live match results for the humans and Ousia model, respectively. Humans and models have considerably different trends in answer accuracy. Human accuracy on both regular and adversarial questions rises quickly in the last half of the question (curves in Figure 6.6). In essence, the “give-away” clues at the end of questions are easy for humans to answer.

On the other hand, models on regular test questions do well in the *first half*, i.e., the “difficult” clues for humans are easier for models (*Regular Test* in Figure 6.7). However, models, like humans, struggle on adversarial questions in the first half.

⁷Videos available at <http://trickme.qanta.org>.

	Adversarial	Regular
Unigram overlap	0.40	0.37
Bigram overlap	0.08	0.05
Longest n -gram overlap	6.73	6.87
Average NE overlap	0.38	0.46
IR Adversarial	0.35	
RNN Adversarial	0.44	
Total Words	107.1	133.5
Total NE	9.1	12.5

Table 6.2: The adversarially authored questions have similar n -gram overlap to the regular test questions. However, the overlap of the named entities (NE) decreases for IR Adversarial questions.

6.6 What Makes Adversarially-authored Questions Hard?

This section analyzes the adversarially authored questions to identify the source of their difficulty.

6.6.1 Quantitative Differences in Questions

One possible source of difficulty is data scarcity: the answers to adversarial questions rarely appear in the training set. However, this is not the case; the mean number of training examples per answer (e.g., George Washington) is 14.9 for the adversarial questions versus 16.9 for the regular test data.

Another explanation for question difficulty is limited “overlap” with the training data, i.e., models cannot match n -grams from the training clues. We measure the proportion of test n -grams that also appear in training questions with the same answer (Table 6.2). The overlap is roughly equal for unigrams but surprisingly higher for adversarial questions’ bigrams. The adversarial questions are also shorter and have fewer NES. However, the proportion of named entities is roughly equivalent.

One difference between the questions written against the IR system and the ones written against the RNN model is the drop in NES. The decrease in NES is higher for IR adversarial questions, which may explain their generalization: the RNN is more sensitive to changes in phrasing, while the IR system is more sensitive to specific words.

6.6.2 Categorizing Adversarial Phenomena

We next qualitatively analyze adversarially authored questions. We manually inspect the author edit logs, classifying questions into six different phenomena in two broad categories (Table 6.3) from a random sample of 100 questions, double counting questions into multiple phenomena when applicable.

Composing Seen Clues	15%
Logic & Calculations	5%
Multi-Step Reasoning	25%
Paraphrases	38%
Entity Type Distractors	7%
Novel Clues	26%
Total Questions	1213

Table 6.3: A breakdown of the phenomena in the adversarially authored dataset.

6.6.3 Adversarial Category 1: Reasoning

The first question category requires reasoning about known clues (Table 6.4).

Composing Seen Clues: These questions provide entities with a first-order relationship to the correct answer. The system must triangulate the correct answer by “filling in the blank”. For example, the first question of Table 6.4 names the place of death of Tecumseh. The training data contains a question about his death reading “though stiff fighting came from their Native American allies under Tecumseh, who died at this battle” (The Battle of the Thames). The system must connect these two clues to answer.

Logic & Calculations: These questions require mathematical or logical operators. For example, the training data contains a clue about the Battle of Thermopylae: “King Leonidas and 300 Spartans died at the hands of the Persians”. The second question in Table 6.4 requires adding 150 to the number of Spartans.

Multi-Step Reasoning: This question type requires multiple reasoning steps between entities. For example, the last question of Table 6.4 requires a reasoning step from the “I Have A Dream” speech to the Lincoln Memorial and then another reasoning step to reach Abraham Lincoln.

6.6.4 Adversarial Category 2: Distracting Clues

The second category consists of circumlocutory clues (Table 6.5).

Paraphrases: A common adversarial modification is to paraphrase clues to remove exact n -gram matches from the training data. This renders our IR system useless but also hurts the neural models. Many of the adversarial paraphrases go beyond syntax-only changes (e.g., the first row of Table 6.5).

Question	Prediction	Answer	Phenomenon
This man, who died at the Battle of the Thames, experienced a setback when his brother Tenskwatawa’s influence over their tribe began to fade.	Battle of Tippecanoe	<u>Tecumseh</u>	Composing Seen Clues
This number is one hundred fifty more than the number of Spartans at Thermopylae.	Battle of Thermopylae	<u>450</u>	Logic & Calculations
A building dedicated to this man was the site of the “I Have A Dream” speech.	Martin Luther King Jr.	<u>Abraham Lincoln</u>	Multi-Step Reasoning

Table 6.4: The first category of adversarially authored questions consists of examples that require reasoning. Answer displays the correct answer (all models were incorrect). For these examples, connecting the training and adversarially authored clues is simple for humans but difficult for models.

Entity Type Distractors: Whether explicit or implicit in a model, one key component for QA is determining the answer type of the question. Authors take advantage of this by providing clues that cause the model to select the wrong answer type. For example, in the second question of Table 6.5, the “lead-in” clue implies the answer may be an actor. The RNN model answers Don Cheadle in response despite previously seeing the Bill Clinton “playing a saxophone” clue in the training data.

Novel Clues: Some adversarially authored questions are hard not because of phrasing or logic but because our models have not seen these clues. These questions are easy to create: users can add *Novel Clues* that—because they are not uniquely associated with an answer—confuse the models. While not as linguistically interesting, novel clues are not captured by Wikipedia or QB data, thus improving the dataset’s diversity. For example, adding clues about literary criticism (Hardwick, 1967; Watson, 1996) to a question about Lillian Hellman’s The Little Foxes: “Ritchie Watson commended this play’s historical accuracy for getting the price for a dozen eggs right—ten cents—to defend against Elizabeth Hardwick’s contention that it was a sentimental history.” Novel clues create an incentive for models to use information beyond past questions and Wikipedia.

Novel clues have different effects on IR and neural models: while IR models largely ignore them, novel clues can lead neural models astray. For example, on a question about Tiananmen Square, the RNN model buzzes on the clue “World

Set	Question	Prediction	Phenomenon
Train	Name this sociological phenomenon, the <u>taking of one’s own life.</u>	<u>Suicide</u>	Paraphrase
Adv	Name this <u>self-inflicted method of death.</u>	<u>Arthur Miller</u>	
Train	Clinton played the <u>saxophone on The Arsenio Hall Show.</u>	<u>Bill Clinton</u>	
Adv	He was edited to appear in the film “Contact”... For ten points, name this American president who played the <u>saxophone on an appearance on the Arsenio Hall Show.</u>	<u>Don Cheadle</u>	Entity Type Distractor

Table 6.5: The second category of adversarial questions consists of clues that are present in the training data but are written in a distracting manner. *Training* shows relevant snippets from the training data. *Prediction* displays the RNN model’s answer prediction (always correct on Training, always incorrect on Adversarial).

One of these concepts ... a **Hyperbola** is a type of, for ten points, what shapes made by passing a **plane** through a namesake solid, ~~that also includes the **ellipse, parabola?**~~
 whose area is given by one-third Pi r squared times height?
 Prediction: Conic Section (✓) → Sphere (✗)

Figure 6.8: The interpretation successfully aids an attack against the IR system. The author removes the phrase containing the words “ellipse” and “parabola”, which are highlighted in the interface (shown in bold). In its place, they add a phrase which the model associates with the answer Sphere.

Economic Herald”. However, adding a novel clue about “the history of shaving” renders the brittle RNN unable to buzz on the “World Economic Herald” clue that it was able to recognize before.⁸ This helps to explain why adversarially authored questions written against the RNN do not stump IR models.

6.7 How Do Interpretations Help?

This section explores how model interpretations help to guide adversarial authors. We analyze the question edit log, which reflects how authors modify questions given a model interpretation.

A direct edit of the highlighted words often creates an adversarial example (e.g., Figure 6.8). Figure 6.9 shows a more intricate example. The left plot shows the *Question Length*, as well as the position where the model is first correct (*Buzzing Position*, lower is better). We show two adversarial edits. In the first (1), the author

⁸The “history of shaving” is a tongue-in-cheek name for a poster displaying the hirsute leaders of Communist thought. It goes from the bearded Marx and Engels, to the mustachioed Lenin and Stalin, and finally the clean-shaven Mao.

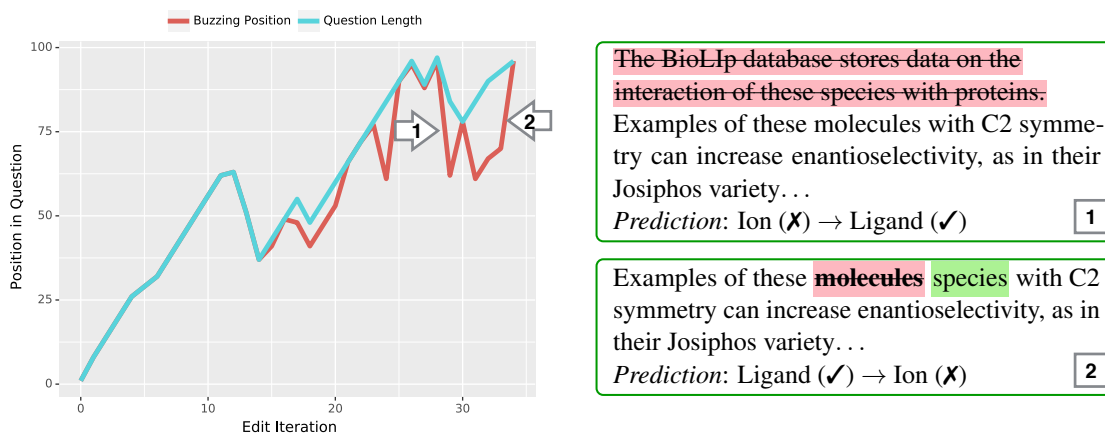


Figure 6.9: The *Question Length* and the position where the model is first correct (*Buzzing Position*, lower is better) are shown as a question is written. In (1), the author makes a mistake by removing a sentence that makes the question easier for the IR model. In (2), the author uses the interpretation, replacing the highlighted word (shown in bold) “molecules” with “species” to trick the RNN model.

removes the first sentence of the question, which makes the question *easier* for the model (buzz position decreases). The author counteracts this in the second edit (2), where they use the interpretation to craft a targeted modification which breaks the IR model.

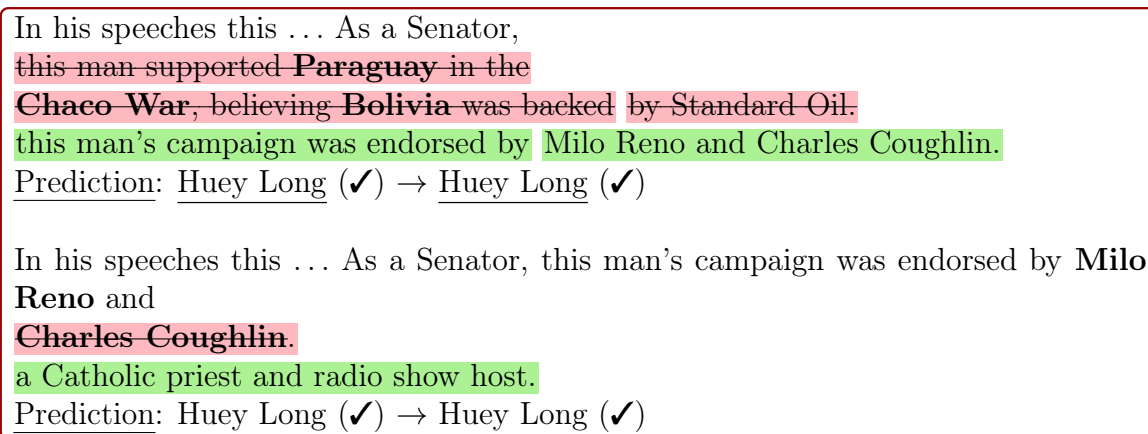


Figure 6.10: A failed attempt to trick the neural model. The author modifies the question multiple times, replacing words suggested by the interpretation, but is unable to break the system.

However, models are not always this brittle. In Figure 6.10, the interpretation fails to aid an adversarial attack against the RNN model. At each step, the author uses the highlighted words as a guide to edit targeted portions of the question yet fails to trick the model. The author gives up and submits their relatively non-adversarial question.

6.7.1 Interviews With Adversarial Authors

We also interview the adversarial authors who attended our live events.⁹ Multiple authors agree that identifying oft-repeated “stock” clues was the interface’s most useful feature. As one author explained, “There were clues which I did not think were stock clues but were later revealed to be”. In particular, the author’s question about the Congress of Vienna used a clue about “Kraków becoming a free city”, which the model immediately recognized.

Another interviewee was Jordan Brownstein,¹⁰ a national QB champion and one of the best active players, who felt that computer opponents were better at questions that contained direct references to battles or poetry. He also explained how the different writing styles used by each QB author increases the difficulty of questions for computers. The interface’s evidence panel allows authors to read existing clues which encourages these unique stylistic choices.

6.8 Related Work

New datasets often allow for a finer-grained analysis of a linguistic phenomenon, task, or genre. The LAMBADA dataset (Paperno et al., 2016) tests a model’s understanding of the broad contexts present in book passages, while the Natural Questions corpus (Kwiatkowski et al., 2019) combs Wikipedia for answers to questions that users trust search engines to answer (Oeldorf-Hirsch et al., 2014). Other work focuses on natural language inference, where challenge examples highlight model failures (Wang et al., 2019b; Glockner et al., 2018; Naik et al., 2018). Our work is unique in that we use human adversaries to expose model weaknesses, which provides a diverse set of phenomena (from paraphrases to multi-hop reasoning) that models cannot solve.

Other work puts an adversary in the data annotation or postprocessing loop. For instance, Dua et al. (2019) and Zhang et al. (2018b) filter out easy questions using a baseline QA model, while Zellers et al. (2018) use stylistic classifiers to filter language inference examples. Rather than filtering out easy questions, we instead use human adversaries to generate hard ones. Similar to our work, Ettinger et al. (2017) use human adversaries. We extend their setting by providing humans with model interpretations to facilitate adversarial writing. Moreover, we have a ready-made audience of question writers to generate adversarial questions.

The collaborative adversarial writing process reflects the complementary abilities of humans and computers. For instance, “centaur” chess teams of both a human and a computer are often stronger than a human or computer alone (Case, 2018). In Starcraft, humans devise high-level “macro” strategies, while computers are superior at executing fast and precise “micro” actions (Vinyals et al., 2017). In NLP, computers aid simultaneous human interpreters (He et al., 2016a) at remembering forgotten information or translating unfamiliar words.

⁹Interviews at <https://youtu.be/MzM1oNsm8MQ>.

¹⁰https://www.qbwiki.com/wiki/Jordan_Brownstein

Other approaches to adversarial evaluation of NLP models (Section 6.2) typically target one phenomenon (e.g., syntactic modifications) and complement our human-in-the-loop approach. Since the original collection of this adversarial dataset, several other datasets have incorporated a similar adversarial human-in-the-loop mechanism (§2.2.3).

6.9 Conclusion

One of the challenges of machine learning is knowing why systems fail. This work brings together two threads that attempt to answer this question: visualizations and adversarial examples. Visualizations underscore the capabilities of existing models, while adversarial examples—crafted with the ingenuity of human experts—show that these models are still far from matching human prowess.

Our experiments with both neural and IR methodologies show that QA models still struggle with synthesizing clues, handling distracting information, and adapting to unfamiliar data. Our adversarially authored dataset is only the first of many iterations (Ruef et al., 2016). As models improve, future adversarially authored datasets can elucidate the limitations of next-generation QA systems.

While we focus on QA, our procedure is applicable to other NLP settings where there is (1) a pool of talented authors who (2) write text with specific goals. Future research can look to craft adversarially authored datasets for other NLP tasks that meet these criteria. For tasks without access to experts, subsequent work in a similar spirit has shown centaur authorship to still be helpful (§2.2.3). In next chapter we also discuss the role of adversarial authorship as one (very useful) tool in the dataset construction toolbox.

Throughout this thesis we improve QA evaluation: this chapter focuses on improving evaluation data so that it more robustly tests QA models for intelligent behavior. In the next chapter, we combine ideas from adversarial centaur authorship with the IRT model developed in Chapter 4 and propose future work to further guide human authors towards writing more discriminative questions. While building towards this future work, we describe a larger view of dataset construction and position centaur authorship as one tool in a greater toolbox for creating better evaluation data.

Chapter 7: Conclusion

The proliferation of competing articulations, the willingness to try anything, the expression of explicit discontent, the recourse to philosophy and to debate the fundamentals, all these are symptoms of a transition from normal to extraordinary research.

Thomas S. Kuhn on Paradigm Shifts
The Structures of Scientific Revolutions

We began this thesis by introducing a new conceptual framework for understanding two branches of QA that have similar yet distinct goals (Chapter 2). The first centers on satisfying users of information-seeking systems (Cranfield paradigm) while the second aims to build QA systems that exhibit intelligent behavior (Manchester paradigm). At the core of this framework is exploring: for what purpose are we building QA datasets, models, and evaluations?

Although these goals overlap, they motivate very different research agendas. For example, in curiosity-driven information-seeking (Chapter 3), user satisfaction goes beyond whether an answer is correct, but to whether the answer inspires user engagement through followup inquiry. This is quite different from Quizbowl where QA systems (or humans) should yield the correct answer, for the correct reason (Chapter 5). Neither paradigm is superior to the other, but we should recognize that their goals are different. For example, adversarially authored questions (Chapter 6) are based on the (Manchester) idea that QA models should correctly answer conceptually similar questions, even if worded differently—otherwise, it would indicate a lack of understanding.¹

After identifying this paradigm difference, we explore ways to improve QA evaluations through better data and formats. In particular, one problem this thesis addresses is that since current QA evaluations are static and unchanging, they are less discriminative. Our improvements are grounded in the idea that as QA models continue to improve, that evaluations should have built-in mechanisms for retaining the ability to distinguish between better and worse models. Quizbowl, for example, retains discriminative ability through its incremental format (Chapter 5). Rather than the binary correct/wrong distinction of other tasks, QB is unique in that questions—by construction—test for multiple ability levels so take longer to become “stale.” In cases where we do not control the “freshness” of examples, we advocate using testing methodologies like Item Response Theory to dynamically

¹However, answering correctly does not necessarily imply understanding.

score models (Chapter 4). However, these ideas are only a beginning in rethinking how evaluations can best support QA research.

Just as TREC shaped the direction of IR research (Voorhees et al., 2005), shared benchmark tasks have shaped the course of machine learning and NLP research (Dotan and Milli, 2020). To make an analogy, shared tasks are akin to large and expensive shared experiments, instrumentation, and laboratories in the physical sciences like the Human Genome Project, telescopes (e.g., Hubble and Kepler), particle colliders such as the LHC, and the International Space Station. These shared resources set the course of research for many years at a time and reflect the scientific priorities and values of their creators. As an example, the investment in QA within TREC “generated a resurgence of research on question answering” (Voorhees et al., 2005, p. 12). Taking a wider view, arguably the Cranfield experiments (Cleverdon, 1967) began a Kuhnian paradigm shift (Kuhn, 2012)² in IR and eventually NLP that was subsequently accelerated by TREC. A similar shift occurred in machine learning more broadly with “the unreasonable effectiveness of data” (Halevy et al., 2009) which was further accelerated by ever-growing datasets (Sun et al., 2017) and computational resources (Kaplan et al., 2020).

However, there is increasing awareness that these successes are not without their dangers, whether it is in NLP (Hovy and Spruit, 2016; Bender et al., 2021) or adjacent fields like computer vision (Birhane and Prabhu, 2021). Across both of these fields, the dangers and harms include, but are certainly not limited to propagating the racial (Blodgett et al., 2016; Buolamwini and Gebru, 2018; Merullo et al., 2019) and gender (Cao and Daumé, 2020) biases of society (Friedman and Nissenbaum, 1996). This is most prominently seen in the creation of the ACM Conference on Fairness, Accountability and Transparency in 2018, along with the veritable wealth of work in the analysis, interpretation, and evaluation of neural models (Belinkov and Glass, 2019). Thus, we are arguably at the precipice of another Kuhnian paradigm shift towards evaluation that looks beyond optimizing headline benchmark numbers in chase of state-of-the-art effectiveness (Linzen, 2020), and towards a paradigm that incorporates the use and impact of NLP models (Mitchell et al., 2019; Blodgett et al., 2020) into evaluations. Next, we propose how shared benchmark tasks (leaderboards) can fuse the ideas of this thesis with these values to improve QA evaluation.

In both industrial practice (Holstein et al., 2019) and academic benchmark tests, a potent place to intervene is in the construction of datasets and benchmark tasks since this often sets the research agenda for many years thereafter and how data is curated has substantial real-world impacts (Rogers, 2021). For example, the first iteration of the EfficientQA shared task (Min et al., 2021) explicitly incorporated efficiency as a core value (Schwartz et al., 2020) which yielded substantially

²Thomas Kuhn’s concept of paradigm shifts posits that science proceeds in four phases: (1) normal, incremental science under an established paradigm (e.g., Newtonian Mechanics), (2) in an attempt to explain significant anomalies in the current paradigm, extraordinary research rapidly pushes the boundaries of science forward under a new paradigm (e.g., General Relativity), (3) the new and old paradigms are debated for which “should in the future guide research problems” (Kuhn, 2012, p. 157), and (4) the new paradigm becomes the dominant paradigm.

smaller systems (Lewis et al., 2021). The remainder of this thesis outlines two larger directions for future work and several extensions. The first two directions build on the ideas of dynamically updating evaluation (Chapters 4 and 6) and rethinking how various stakeholders should engage with leaderboards. The subsequent extensions focus on adapting additional IRT methodology into NLP evaluations.

7.1 Future Work: Living Evaluations

A central theme of this thesis is improving the discriminative power evaluations through additional annotation (Chapter 3), dynamically weighting examples (Chapter 4), incremental and pyramidal formats (Chapter 5), and adversarial evaluation data (Chapter 6). These are all in service of a greater vision that evaluations should—from the onset—be built to evolve as progress is made on the task(s) of interest.

Conceptually, these and other methods are tools in a larger toolbox of ways to improve the efficacy of evaluations. Taking this analogy further, our toolbox is somewhat disorganized. Although each particular tool may bias the data in one way or another, they are all useful since they still expand the scope of natural language phenomena that we test. We envision a data collection and evaluation framework that combines Item Response Theory and adversarial annotation in a format where examples test one or more levels of knowledge.

7.1.1 Incorporating Content Models into IRT Models

The first step towards this framework builds on the idea of IRT for QA evaluation by incorporating a content model. A glaring drawback of standard IRT methods is that to infer example properties such as difficulty or discriminability, they require scored predictions from at least a few models. Suppose we obtain a new example that is nearly identical to an existing example. To IRT, these examples have no relationship until we infer it by seeing that models perform similarly on both examples. The missing ingredient is a content model to tell us that these examples are similar and thus should have similar IRT properties.

For inspiration, we look towards political science where Ideal Point Models (Poole and Rosenthal, 1985)—which are mathematically equivalent to IRT—are used to predict the political leanings of legislators (“subjects”) based on their voting records (“responses”) on bills (“items”). As in our case, political scientists also endeavour to incorporate the textual similarities of bills to infer voting patterns (Gerish and Blei, 2011; Nguyen et al., 2015; Kraft et al., 2016).

To combine content models with IRT models, we propose two extensions to the IRT model in Chapter 4: (1) integrating example metadata (Card et al., 2018) like topical category and (2) incorporating a topic model (Blei et al., 2003). As with any Bayesian model, the primary changes will be to the generative story (§4.2). For metadata like topical category, we will assume an uninformative prior distribution and make previously entirely latent variables like difficulty and skill depend on

the metadata prior. For example, each topical category could be associated with independent mean difficulties, which would allow the model to associate particular topical categories with different difficulty distributions (e.g., perhaps literature questions are more difficult).

The second extension builds on this concept by integrating a topic model into the IRT model. Specifically, we link the IRT and topic models through the latent properties of items like difficulty and discriminability. Since it is reasonable to presume that specific concepts have intrinsic difficulties, the latent difficulty of a particular example should inform both which topic is selected and the words used in that example. Thus, as before the generative story first draws the properties and then draws the values for dependent random variables—in this case, the responses and topic model features. In both extensions, standard methods for inferring distributions of new “documents” (examples) could be applied to solve the problem that standard IRT treats each example as an entirely new data point. However, they also provide the additional benefit of improving the interpretability of IRT models, which we use next to guide example creation.

7.1.2 Guiding Example Creation

Perhaps the most ambitious version of an IRT-based tool for creating examples is one that does it all: users specify the desired difficulty of a question, and the tool outputs the example. However, a major challenge with this approach is that generating coherent text is a hard problem in its own right.

We avoid this challenge and instead propose a tool that uses content-based IRT models to guide human annotators in creating new examples. During annotation, the content-based IRT model could infer example characteristics (e.g., topic category) that increase the likelihood of matching a pre-specified difficulty. For example, perhaps questions of a particular topic or that mention particular words or entities tend to be easy. As with centaur authoring (Chapter 6), this combines the strengths of machines like inferring distributional statistics with the strengths of humans like authoring questions.

For this task, the metadata-based and topic-based views have complementary strengths. The topic-based IRT models provide finer-grained word-level guidance but have the drawback that they can only suggest words seen during training. Perhaps history questions mentioning “Taíno” tend to be more difficult, but if a model does not see it during training it cannot suggest it. To cover for this weakness, metadata models could specify properties like the topical category or the entity type of the answer, which could help human annotators narrow down the type of question to write. Although the IRT model synthesizes which types of questions leaderboard models find easy or hard, it is not a direct replacement for having a model-in-the-loop to test if questions are too easy (and thus potentially useless). In Chapter 6, although we employed multiple models, they were not used at the same time; in this followup work, we propose using multiple models from the leaderboard for adversarial checking, which would also provide feedback to the IRT model. Although we certainly should collect new examples as leaderboards evolve, we should not so

readily commit underperforming examples to the deep.

7.1.3 Tender Loving Care for Underperforming Examples

An important step towards improving evaluations is not simply collecting new datasets to fix the problems of existing datasets, but fixing problems in existing datasets. Towards this goal, methods like IRT and those that derive this from training dynamics (Swayamdipta et al., 2020; Pleiss et al., 2020) identify “bad examples.” In a living evaluation, “bad examples” are sent to annotators where they are either discarded as unfixable or fixed through editing. A third possibility exists though, an example could be perfectly reasonable, just too easy.

7.1.4 Multi-Examples

For examples that are “too easy,” it is difficult to know whether the underlying linguistic, knowledge, or reasoning skill being tested is genuinely too easy or if simply this particular instance is too easy. An elegant solution to this problem is to test the concept of interest multiple times and only mark it correct if all variations are correct. As Chapter 5 discusses, QB questions test knowledge multiple times, with unique aspects that each subsequent test is easier and easier. QB’s version of this idea is that if a model answers a question correctly at certain position, it should *also* answer the question correctly at all subsequent points; this intuition is encoded in the expected wins metric (§5.7.1). Gardner et al. (2020a) introduces a similar idea where expert annotators create multiple versions of test examples, and examples are only marked correct if all sub-examples are correct. Our proposal is rather than selecting these examples randomly, that we prioritize transforming weak examples into *multi-examples*.

7.1.5 Effects of Continually Updating Evaluations

A substantial “drawback” of living evaluations is that precisely because they are ever-changing, it is more challenging to compare models across time. Perhaps a current state-of-the-art model today is no longer state-of-the-art tomorrow once bad examples are fixed or transformed into multi-examples.

First, we can mitigate this problem by requiring that *runnable* models are submitted to leaderboards. This is already done by several tasks such as (e.g., SQuAD and NaturalQuestions) and should be adopted more broadly (Crane, 2018). Since runnable models are submitted, leaderboard organizers can rerun models as evaluation data change. A secondary benefit of this requirement is improved reproducibility; reproducing evaluation predictions and scores becomes trivial.

Second, perhaps a continually updating evaluation will discourage research that solely optimizes metrics without providing insight into why a method works (Linzen, 2020). Even in industry applications where effectiveness may hold more sway, it is still important to understand the contributions and limitations of methods, otherwise we risk unpredictable and potentially harmful deployments (Buolamwini and

[Gebru, 2018](#); [Wallace et al., 2019a](#); [Carlini et al., 2020](#)). We will soon take this idea further and propose ways to change leaderboards so that they naturally encourage deeper inspection of data and models at both single-model scale and across all models.

7.1.6 Model Cards

Several authors advocate for more detailed reporting of model characteristics ([Mitchell et al., 2019](#)) and associated training data ([Gebru et al., 2018](#); [Bender and Friedman, 2018](#)). Their motivation—which we whole-heartedly agree with—is to improve transparency and accountability, and documenting models when submitting to leaderboards should be required. Although to some this may seem onerous and cumbersome, we can make it more attractive to model developers by providing something in return. Thus far, we have discussed ways to incorporate information about the content of examples into IRT, but have not considered integrating model properties.

In IR, Query Performance Prediction ([Carmel and Yom-Tov, 2010](#)) infers the characteristics of models that are most predictive of system effectiveness for particular inputs (queries). As a starting place, self-reported data from model cards (e.g., architecture, number of parameters, datasets used during training) could provide information for IRT or other similar methods to use. This information could be incorporated into the IRT generative story or analyzed as with standard statistical methods like inspecting feature importances of a linear model. By requiring runnable models though, we also open the door to more sophisticated analysis that might use aggregate data like score distributions or robustness to newly annotated examples. If leaderboards helped model developers determine which characteristics—across all submitted models—are most effective then it may provide intrinsic incentive for good model documentation. Next, we further develop the idea that leaderboards should do more than rank models according to one or more metrics.

7.2 What More Should Leaderboards Do?

Taking a broader perspective, leaderboards are truly shared tasks in disguise. When benchmarks like SQuAD are adopted by a community they implicitly become shared tasks. Shared tasks—whether they are explicitly or implicitly created—have been instrumental to progress to NLP in areas like speech recognition ([Pallett, 2003](#)), information retrieval ([Voorhees et al., 2005](#)), and question answering. [Nissim et al. \(2017\)](#) succinctly describe NLP shared tasks as revolving “around two aspects: research advancement and competition. [They] see research advancement as the driving force and main goal behind organizing them. Competition is an instrument to encourage and promote participation. However, just because these two forces are intrinsic to shared tasks does not mean that they always act in the same direction.”

Leaderboardized shared tasks have swung too far towards optimization of target effectiveness metrics. While this has certainly driven rapid progress in NLP

modeling, as [Bender et al. \(2021\)](#) argues,

Where much effort has been allocated to making models (and their training data) bigger and to achieving ever higher scores on leaderboards often featuring artificial tasks, we believe there is more to be gained by focusing on understanding how machines are achieving the tasks in question.

Sadly, sometimes it is *because* of competition that shared task participants who do not win feel describing their systems or insights is not worth their time ([Parra Escartín et al., 2017](#); [Pedersen, 2019](#)). At the same time, while there has been a great interest in the analysis of NLP models ([Belinkov and Glass, 2019](#)) and providing tooling to test robustness ([Goel et al., 2021](#)), there has not yet been an effort to close the loop and bringing these elements into benchmark tasks and thus into the view of model developers ([Linzen, 2020](#)). We argue that leaderboards should play a central role in re-aligning their implementation with their original intent: to advance research.

Our re-imagining of leaderboards takes inspiration from computer visualization by identifying (1) the stakeholders of leaderboards, (2) the goals of these stakeholders expressed as abstract tasks, and (3) how leaderboard components contribute towards these tasks ([Munzner, 2014](#), p. 43–45). We will first introduce the stakeholders and then discuss how each of their goals are satisfied in one or more of three leaderboard visualizations. These three visualizations include a ranking-centric view, a model-centric view, and an example-centric view. By re-designing leaderboards with explicit goals in mind, we can use software tooling as a means to lower the barriers in achieving those research goals ([Myers et al., 2000](#)).

7.2.1 Stakeholders

[Ethayarajh and Jurafsky \(2020\)](#) take a similar user-centric approach to define the utility of leaderboards, but limit their analysis to NLP practitioners who submitted their models to leaderboards. Here, we take a wider view inspired by value-sensitive design ([Friedman et al., 2008](#)) and identify several direct and indirect leaderboard stakeholders, including the public, academia, industry, participants, and task organizers.

The Public To the press and thus the public, leaderboards are a prominent means to communicate progress in AI technology but carry the risk of overstating the capabilities and understating the limitations of such technology. Focusing on the public as a stakeholder, leaderboards should faithfully communicate the progress actually made, emphasize its limitations, and convey uncertainty in each of these. Unsurprisingly, to this stakeholder, the ranking view will have the most importance (§7.2.2).

Academia The role of leaderboards in academia is—for better or worse—unmistakable: obtaining state-of-the-art effectiveness is often a means to, if not an implicit requirement to publish research papers at top-tier venues. Consequently, this motivates

some participants to submit models as a step towards publishing papers that ultimately earn reputational prestige. With the growth of NLP and adjacent areas though, this model is unsustainable, and the living evaluation we propose muddies the entire concept of “state-of-the-art.” Taking a step back, in any re-design of a rankings view, we should recognize that when optimization of a specific metric becomes the primary goal—as opposed to the original task—that leaderboards are vulnerable to the effects of Goodhart’s Law (Strathern, 1997). In their current form, one value of benchmarks is in unifying evaluation data so that approaches can be more easily compared—a goal that does not require the selection of specific, possibly value-laden metrics (Dotan and Milli, 2020). We will argue that they should also help identify fertile research areas by highlighting characteristics of difficult examples (§7.2.4) and properties of effective models (§7.2.3).

Industry Where academia is famously data-poor, industry is data-rich, and by providing suitable data for tasks of interest, industry shapes the course of academic research. For example, IR datasets like AOL search queries (Pass et al., 2006), TREC tasks using commercial search engine queries, MS MARCO, NaturalQuestions, and others encourage research where academic and industry interests intersect. Along similar lines, government programs like IARPA often contribute funding, datasets, and evaluation metrics to advance research of interest to the US government.³ These stakeholders are differentiated from academia in that they have a financial incentive in ensuring that progress on target metrics translates to advances that ultimately improve products (e.g., the Netflix Prize §7.2.2). Thus, to this stakeholder, the ability to robustly measure progress resulting in practical impacts is of great value.⁴

Participants Next are the participants of benchmarks tasks. Since we have already discussed leaderboards as SOTA rubber stamps, here we focus on participants seeking to improve their models. Somewhat paradoxically, leaderboards are perhaps *least* supportive of this group; aside from listing effectiveness metrics like accuracy, they—generally—do not provide any other features. The model-centric (§7.2.3) and example-centric (§7.2.4) leaderboard views will be designed to help this group find ways to improve their models.

Organizers Lastly, task organizers are primarily concerned with managing the lifecycle of the shared task.⁵ To this group, providing insight into the health of the benchmark and how to fix issues is key. Towards this goal, features that aid in identifying and fixing bad examples are helpful. Therefore, to this group, the example-centric view (§7.2.4) will have the most value. Next, we describe the three proposed leaderboard webpage views and how they each meet stakeholder goals.

³The IARPA MATERIAL program supports work in low-resource, multilingual information retrieval.

⁴For example, MATERIAL evaluation sets are regularly updated, with each using a new low-resource language to encourage work that is not overly reliant on any specific language pair.

⁵Here we consider goals independent of their academic or industry role(s).

7.2.2 Ranking View

The standard role of leaderboards is to facilitate the comparison of models through aggregated metrics representative of the shared task’s goal(s). In most cases, there is a single metric of interest such as accuracy, but even when single metrics reflect the eventual end task as in many industry-sponsored shared tasks, they seldom integrate the full spectrum of concerns in the end task. In the Netflix prize (Bennett and Lanning, 2007), both the winning entry and followup made heavy use of ensemble methods (Koren, 2009) which were too difficult to scalably and efficiently directly use in production deployments (Amatriain and Basilico, 2012). That said, the competition was successful in producing research that led Netflix to incorporate successful innovations into production deployments. The missing element of the competition is that not all the values of interest were incorporated—in this case simplicity and efficiency. The first goal we focus on is to reward many types of research progress by increasing the diversity of metrics used in leaderboards. First though, we need to identify at least a few metrics to start with.

Metrics To start, we propose that leaderboards should integrate metrics for effectiveness, robustness, generalizability, efficiency, and fairness. Since effectiveness metrics are status quo, we will not discuss these beyond noting they should be appropriate to the end task. Towards motivating efficient AI, the GreenAI initiative (Schwartz et al., 2020) recommends counting the total number of floating-point operations as a proxy for measuring carbon impact. However, there is plenty of room for alternatives; for example, the EfficientQA (Min et al., 2021) shared task focused on decreasing the memory footprint of models which is important for mobile deployments (Sun et al., 2020). For fairness, we can begin with demographic parity (Barocas et al., 2019, p. 9–10),⁶ but there are also many alternatives, and each leaderboard should think critically about which to use. While incorporating these metrics is not the only way or even the best way to integrate alternative values into evaluations, it represents a first step.

To integrate a robustness metric, we should make use of the aforementioned idea of *multi-examples* (§7.1.4). We initially presented this concept as a way to augment existing examples through human annotation and requiring all sub-examples to be correct for the full example to be correct. This also naturally lends itself as the basis for a robustness metric; for example, the ratio of correct to incorrect multi-examples could be a robustness metric. In addition to human-created multi-examples, we should also employ techniques that use rule-based methods to generate additional examples (Jia and Liang, 2017; Ribeiro et al., 2018). Along similar lines, behavioral testing (Ribeiro et al., 2020) could also contribute towards a robustness score while helping to identify capability-based tests.

Generalizability is already a topic of interest in leaderboards through multi-task leaderboards. Examples include SUPERGLUE (Wang et al., 2019a) and MRQA (Fisch et al., 2019). The challenge in these multi-task benchmarks is aggregating multiple

⁶For example, effectiveness across a pre-defined set of partitions should be the same (e.g., accuracy for questions about men versus women).

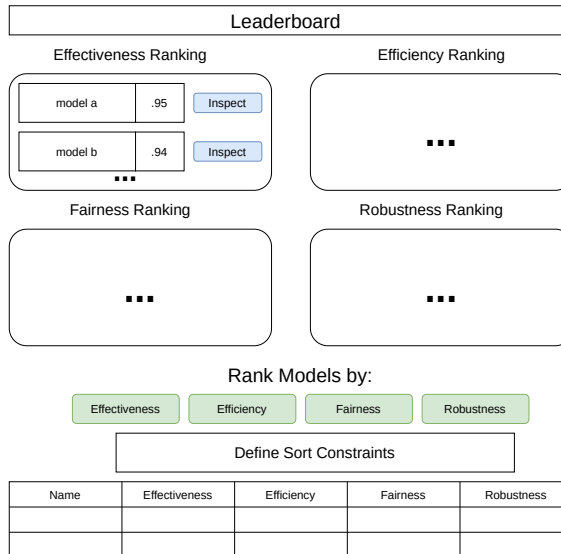


Figure 7.1: The ranking view and landing page of leaderboards should convey that the community values multiple types of progress. By highlighting the models according to different metrics, the ranking view de-emphasizes the importance of any single metric and encourages more thought into deciding what does the concept of “best model” mean. Lastly, rather than highlight only the highest-scoring models, we shift towards highlighting clusters of comparable models; for example, group all models whose scores are not statistically-speaking different.

effectiveness metrics into a single, representative number. The most common approach averages task scores, but this implicitly assumes that each task is equally easy. As alluded to earlier (§4.9), dynamically weighting scores by difficulty is a strength of IRT models from Chapter 4. Preliminary experiments (Appendix B.6) show this has promise, but that correlating task effectiveness with dimensions is challenging. Conceptually, the problem is that the generative model places no prior on the distribution of task difficulty in latent space. As future work, we propose changing the generative story to use Dirichlet priors so that difficulty is more likely to be concentrated in fewer dimensions (Teh et al., 2006). Additionally, a Bayesian domain adaptation approach could model per-task parameters separately while tying them together through a hierarchical Bayesian prior (Finkel and Manning, 2009). Hopefully, this will improve the interpretability of and thus confidence in using IRT skill parameters as a way to aggregate multi-task effectiveness.

Table of Ranked Models Ultimately, the ranking view needs to contain a table showing metrics for each model. The question is how to rank models in a way that does not re-introduce the problems we are trying to mitigate such as SOTA-chasing at the expense of everything else (Rogers, 2019). We propose a middle-ground approach to designing the ranking view.

First, we depart from the standard layout of leaderboards—displaying a ranked list of models by a metric—and instead show four groups of models (assuming the

leaderboard evaluates effectiveness, efficiency, fairness, and robustness). This change serves to de-emphasize the importance of any single metric (and its underlying value) and place more emphasis on alternative ways to improve models.

Each group shows the top model according to the associated metric but groups it with all models that are considered comparable (Figure 7.1). For example, a common problem with current leaderboards is that the difference between top-performing models tends to be small, even to the point of being statistically indistinguishable. In our proposed ranking view, statistical significance is one way to determine if models are comparable. For an efficiency-based metric, perhaps model comparability is defined by a pre-specified trade-off between accuracy and computational cost (e.g., a one-point drop in accuracy is acceptable for ten-point improvement in runtime). More generally, users (or task organizers) should consider trade-offs that align with their notions of utility (Ethayarajh and Jurafsky, 2020).

Beyond tabular rankings, we should look towards new ideas in computer visualization. For example, when using mixed utility functions, this information could be conveyed with a hybrid between a table and a bar chart (Gratzl et al., 2013).

To academia and the public, this design emphasizes that there are multiple ways to advance research and the importance of balancing these factors. The secondary benefit of this design is in explicitly integrating uncertainty by identifying which groups of models are comparable. To industry stakeholders, the leaderboard better serves their goals by supporting the means to define utility while rewarding focus on particular aspects of the task (e.g., efficiency versus accuracy). From the organizer’s perspective, rewarding different kinds of progress may also incentivize a wider variety of shared task system descriptions.

7.2.3 Model View

Leaderboards should have a model-centric view whose purpose is to guide model developers in improving their models. Frequently, the development of ML models is an iterative process of diagnosing problems and fixing them until the practitioner is satisfied with the quality of the model (Patel et al., 2008a). Thus, diagnostic tools are a means to “naturally lead developers towards doing the right thing” (Patel et al., 2008a). Depending on which tools we design, we can guide developers in different directions. As with the ranking view, there are many existing visualization techniques for model debugging including interpretations of single models (Wallace et al., 2019c; Wu et al., 2019), pairwise model comparison (Zhang et al., 2019), multi-model comparison (Zhang et al., 2019), attribute-based comparison (Arendt et al., 2021), example-based comparison (Amershi et al., 2015), and others.

Some of these tools should play a role in a model-based view, but a unique aspect of a leaderboard-based tool is access to all other models. An open challenge in this area is exploring options that help compare a submitted model to other models in aggregate (e.g., perhaps by treating other models as an ensemble).

7.2.4 Example View

The third view leaderboards should support centers on identifying patterns in the evaluation data. Since task organizers are concerned with the health of the shared task, this view should aim to identify and help fix inadequacies of the test data.

A first step towards this should include incorporating the IRT methods used to identify bad examples (§4.5), but extend to other ways to identify bad examples. For example, [Swayamdipta et al. \(2020\)](#) use the training dynamics of neural models to identify *hard to learn* examples, which often correspond to labeling errors. Since their method relies on training dynamics, in its standard form, it does not provide characteristics of development or test examples, but it would be easily modified to do so. K-fold validation would allow the development data to be used (temporarily) as training data to produce the desired quality metrics on test data.

The example view should make it easy for organizers or participants to find these examples and validate whether they are flawed. For private evaluation data, this obviously must be done by organizers (or annotators), but this seems like a reasonable cost to pay for better evaluations.

7.3 Future Directions in Item Response Theory for NLP

Thus far, we have briefly covered a few ways to improve IRT for NLP; here, we enumerate two additional directions.

7.3.1 Multidimensional Clustering

A glaring weakness of Chapter 4’s IRT models is its reliance on one-dimensional parameter values. It is quite conceivable that parameters like difficulty and skill could vary along more than one dimension: say math difficulty and literature difficulty. Although we did train multi-dimensional models (§4.2), initial experimentation failed to correlate specific dimensions with example features, so we only reported numerical effectiveness results. An alternative approach to pre-specified features is clustering models or examples by multi-dimensional IRT parameters and doing post-hoc analysis of clusters.

Ideally, multi-dimensional clustering on a multi-task benchmark like MRQA ([Fisch et al., 2019](#)) would show a relationship between clusters and sub-tasks. Unfortunately, initial investigation (Figure 7.2) shows only limited correlation with one of the six MRQA tasks. The main challenge of future work is to investigate why it seems like no particular dimension correlates with specific tasks. Our preliminary hypothesis is that nothing in the generative model encourages skills associated with particular tasks to accumulate in a single dimension versus spread equally across all dimensions. Thus, a modified generative model that uses Dirichlet priors to encourage more meaningful latent skill distributions (§7.2.2) may fix this problem and allow for better empirical investigation of which skills are related to each other.

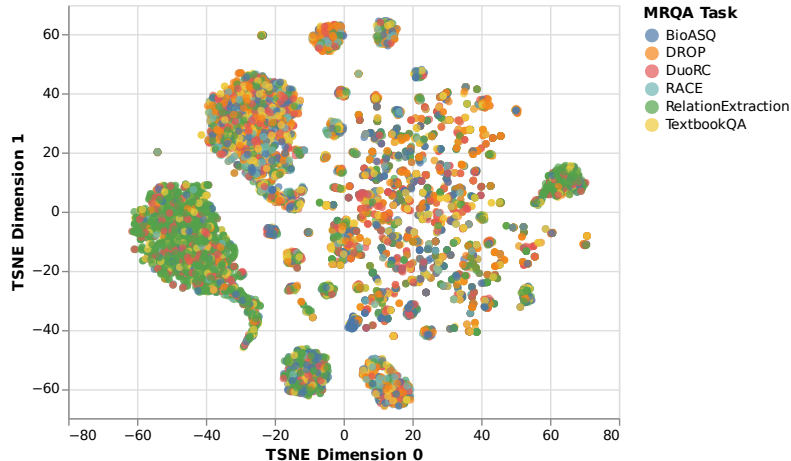


Figure 7.2: In MRQA, TSNE shows a relationship between whether the task is NarrativeQA with respect to multidimensional difficulty and discriminability. The multidimensional IRT model uses six dimensions to match the six MRQA tasks.

7.3.2 Statistical Testing

In IRT, a useful concept is the informativeness of examples towards inferring a subject’s latent skill (Equation (4.2) in §4.4.5). Chapter 4 uses this to adaptively select examples for annotation. One may ask a related question: how much information (and by extension the number of annotations) is required to estimate a subject’s skill to a particular precision with a certain degree of confidence? In human testing, this is important for determining things like how long an exam should be; longer exams provide more information but are more costly to administer.

This question is directly related to statistical testing in NLP to determine whether the difference between two systems is statistically significant or insignificant. Although including such tests in publications is strongly recommended (Dror et al., 2018), most statistical tests in NLP publications only account for variation within whole model runs; however, when evaluation sets are shared, it would be more appropriate to use paired tests that examine how a collection of models score on shared examples. For example, reasonable statistical tests include Student’s t -test, the sign test, McNemar’s test, the Wilcoxon signed-rank test (Wilcoxon, 1945) (previously recommended by Demšar (2006)), Pitman’s permutation test, and the paired bootstrap test. Card et al. (2020) investigate how these and other statistical tests are used in NLP and find that in many cases the tests are under-powered (unable to distinguish between better and worse models). As we will briefly outline, an IRT-based statistical test brings several unique properties that future work should characterize empirically compared to existing tests.

The IRT tests differ in two significant ways from other tests: (1) it does not assume that items are equally informative, and (2) it assumes that the informativeness of items is a function of the subject’s skill θ_j . Under these assumptions, each item provides information about the true value of a subject’s latent skill, and as we accumulate more evidence, the reliability of this estimate improves (Tague-Sutcliffe,

1992). As consequence of skill-dependent information, the statistical error associated with the estimate also varies with skill. In IRT, this error is called the standard error of estimation (SEE) $\sigma(\hat{\theta}|\theta)$ (De Ayala, 2013, p. 30) and is related to the Fisher information

$$I_i(\theta) = \frac{(p'_i)^2}{p_i(1 - p_i)} \quad (7.1)$$

of each item; previously, we used item information to adaptively sample examples for annotation (§4.4.5). For a 2PL model, information

$$I_i(\theta) = \gamma^2 p_i(1 - p_i) \quad (7.2)$$

is maximized when $p_i = (1 - p_i)$. Since Fisher information is additive, the information of the evaluation set is maximal when items have a 50% chance of being responded to correctly. As derived by De Ayala (2013, p. 102), the standard error of estimation

$$\text{SEE}(\theta) = \sqrt{\frac{1}{\sum_i I_i(\theta)}}. \quad (7.3)$$

is computed by accumulating the information gained from each item. Given two subjects X and Y , the probability distribution of score differences

$$N(\theta_Y - \theta_X, \text{SEE}(\theta_X)^2 + \text{SEE}(\theta_Y)^2) \quad (7.4)$$

can be used to compute the probability that the difference in skill is statistically significant; for example, a difference of greater than two standard errors corresponds to an $\alpha \leq .05$ significance level. Future work should investigate how the different assumptions the IRT test makes influence experimental results. Ultimately, perhaps the main challenge in this line of work is in determining which statistical test is “correct,” which is difficult or impossible to determine with real-world data; it is likely the investigation into comparing these tests will include simulations where differences are a priori known to be significant or insignificant.

7.4 Reflections and Synthesis

Throughout this thesis, I have woven a story and believe it would be instructive to explain how I came to that story. As an early PhD student, my first step towards working on QA evaluation was building the QA system that would eventually defeat accomplished trivia players in a live exhibition match. Much as this excited me, it was somewhat unsatisfying that a system built on pattern matching and therefore distinctly not “intelligent” had been sufficient to defeat skilled humans. This led to asking “should we interpret the victory of a—by construction—not intelligent model a success for QA?” and eventually the better question of “how should we define success for QA?” In an effort to not allow machines to “cheat” their way to victory through statistical pattern matching, Chapter 6 and its adversarial questions suggested that if models do not rely on “spurious” correlations, then that is success. In direct contrast, the Curiosity dataset in Chapter 3 defined success as satisfying the

user. The missing ingredient through all these questions is that defining success is a value judgment, and I was attempting to answer this question without considering values.

This realization led me to develop the idea to view QA through the lenses of the Cranfield and Manchester paradigms. Crucially, each of these paradigms defines what it values and by doing so define their own notions of success.

Viewing QA evaluation from the Turing Test inspired Manchester paradigm, the theoretical underpinnings for QB's pyramidal and incremental format became clear. Methodologically speaking, QB tests for multiple, different levels of knowledge understanding which directly incorporated the idea that questions (clues in QB) are not equally informative of ability. This line of thought eventually grew into using IRT for QA evaluation, but was not the only reason for adopting IRT. The other attractive feature of IRT is the additional insight it gives into the data and models in leaderboards. By making leaderboards—as a tool—more helpful (i.e., not just a ranked list of models), perhaps this would accelerate progress towards better QA systems.

In thinking through how leaderboards might be made more helpful, the question of what goals and thus values to optimize for resurfaces. In revisiting the question, it is also crucial to internalize that widely adopted shared tasks often set the goals and focus entire communities of research for multiple years at a time. By identifying stakeholders and what they value, it makes leaderboards more effective in advancing research. And while we are thinking critically about the values integrated into leaderboards, we can also prioritize new values—like efficiency and fairness—and thereby reward research prioritizing those values.

Throughout, the underlying and driving questions in this thesis have been: how do we define “better” and the necessarily dependent question of how to accurately measure that. Towards the goal of building QA systems that exhibit intelligent behavior, this thesis advocates for specific methods to improve the format, data, and scoring in QA evaluations. Although these methods are perhaps specific to improving QA evaluations, the broader message of this thesis is to always think critically about the purpose and goals of computer systems.

Appendix A: Curiosity

This appendix and those that follow are each paired with a particular chapter of this thesis. In these appendices, we primarily include: (1) details important for reproducibility, but that are uninteresting and break the flow of the main text and (2) additional examples or annotations. In this appendix, we provide additional details of the interface, dataset, and models in Chapter 6.

A.1 Components of Dialog Interfaces

This section provides short descriptions and screenshots of every component of the user and assistant dialog interfaces.

A.1.1 User’s Interface

Figure 3.3 shows the interface that we use to sample the user’s prior knowledge of entities related to the topic. To derive a diverse sample, we use Wikipedia page views as a proxy for how well known the entity is. All experiments use the English Wikipedia dump generated on July 23, 2019. We divide entity mentions into ten buckets based on the frequency of page views, and round-robin sample fifteen entities from those buckets. The interface is shown before the user starts chatting with the assistant.

We elicit how “interesting” a user finds each of the assistant’s messages through the like button in Figure 3.4. Only users can “like” a message; the assistant cannot “like” user messages. Users are instructed to “like” messages if they are “interesting, informative and/or entertaining” and “relevant to their topic and/or aspects.” They are specifically instructed not to “like” messages that are devoid of factual content, only express feelings, or only contain greetings or farewells.

Switching Aspect Users are randomly assigned two aspects for each dialog and told to spend time discussing each. The guidelines instruct them to spend at least two turns per topic, but we do not specify any further time requirements. When the user changes aspects, we instruct them to click a button (Figure A.1) to indicate when and which aspect they are switching to. Additionally, this event triggers a reset in the context we use to rank the assistant’s facts.

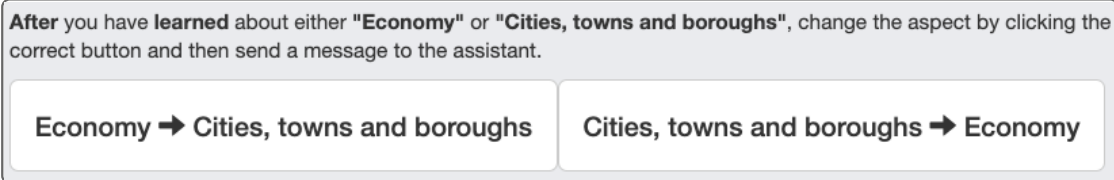


Figure A.1: The user is assigned two aspects about their topic. After they are satisfied with what they have learned about the first aspect, they click a button and switch to the next aspect. While the button click is not communicated to the assistant (the user must send a corresponding message), it resets the fact contextualizer; we observe that without this, too many facts were related to the previous aspect.

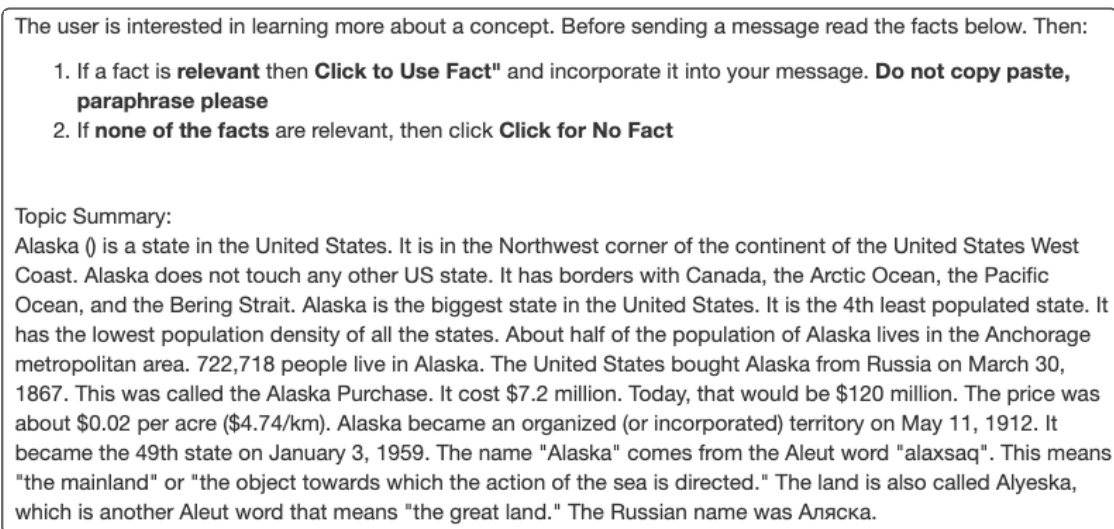


Figure A.2: A short topic description is always visible to the assistant. The goal is to ensure the assistant always has a general understanding of the dialog topic.

A.1.2 Assistant Interface

By design, we intend for most workers to not be familiar in depth with most of the geographic topics. Thus, the most important responsibility of the assistant interface is to provide enough information—without overwhelming them—to be engaging conversational partners. The first interface shown is a short description of the topic from either Simple Wikipedia or the English Wikipedia. This component helps the assistant reach a general understanding of the topic so that they can choose better facts.

The most important component of the assistant interface is their list of available facts. These facts have high textual similarity with the most recent three turns and are broken into three categories: facts related to entities the user knows about (rooted facts), facts related to an aspect (aspect facts), and facts from anywhere on the page (general facts). Feedback from pilot collections showed that six facts was too few which caused a lack of relevant facts, but twelve facts overwhelmed annotators. Thus, we use nine facts so that we can also balance equally across each

type of fact. When composing their reply, the assistant can use any number of facts as in Figure 3.5. To discourage verbatim copying, we disable the paste feature in javascript. We also drop repeatedly unused facts.

A.2 Dialog Act Annotation

To annotate dialog acts, we create a separate annotation interface (Figure A.3). The interface shows one dialog at a time, and the same annotator annotates all the utterances. In addition to the utterances, the interface shows the topic, aspects, and sender of each message. Lastly, we incorporate a “Report Dialog” feature to help identify and remove inappropriate dialogs.

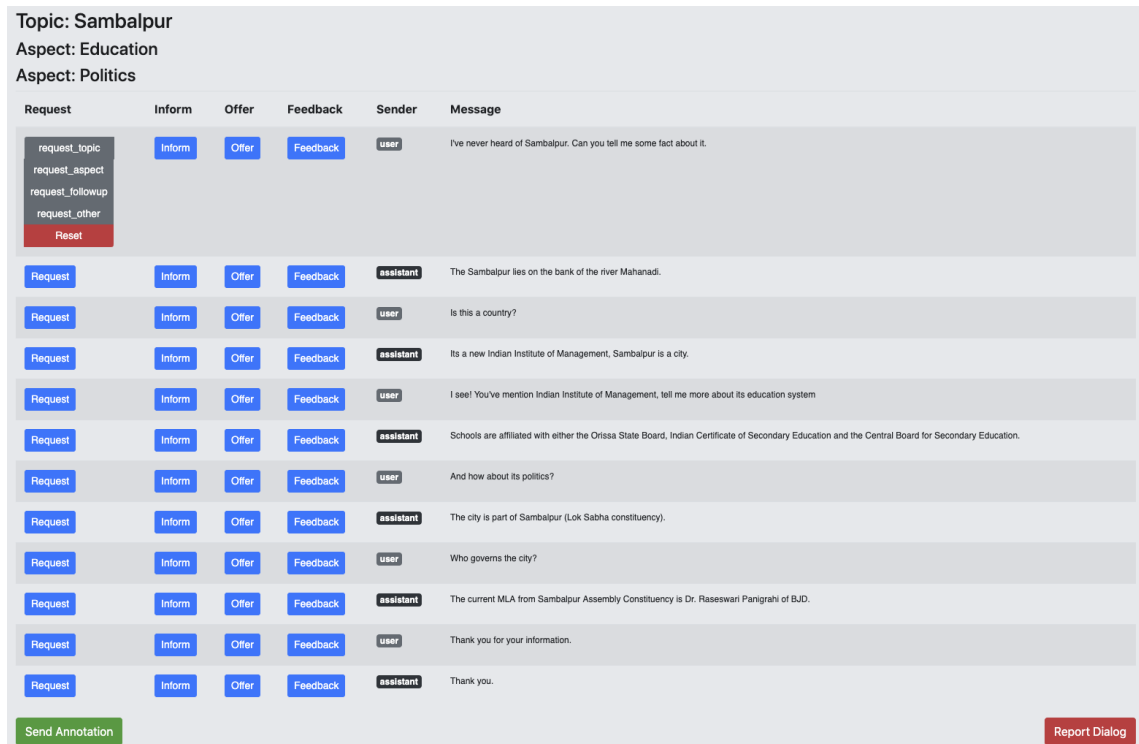


Figure A.3: To annotate dialog acts, we develop an interface that showed each utterance on a separate line. Annotators assign zero or more dialog acts to each utterance using grouped dropdowns.

A.3 Sample Dialogs

Tables A.1 and A.2 show Curiosity dialogs and highlight the dataset’s features. Typos and grammatical errors made by annotators are left unaltered.

A.4 Paraphrase Analysis and Samples

In Section 3.3.1.3, we describe the results of a manual analysis on two hundred and fifty assistant paraphrases. Annotations were completed by the authors and shown in Table 3.3. We break messages into four categories: paraphrases, copies, errors, and unrelated. Paraphrases include messages that incorporate the selected fact and possibly additional information. Copies include verbatim copying, cherry-picked phrases, and trivial contextualizations like replacing an entity with a pronoun.

A.5 Like Prediction Comparison

Like prediction is the one task where CHARM was not the best model. To better understand the differences between the CHARM and BERT model, we randomly sample thirty dialogs in the test set and find assistant messages where the model predictions disagree. Of the 202 assistant messages in the thirty dialogs, the BERT like prediction is correct 174 times (86.1%) and CHARM 170 times (84.2%). The models disagree on eight predictions (3.96%); BERT is correct on six of these and CHARM on the other two. Table A.4 shows the messages, predictions, and labels. Although we might guess at why the predictions differ (e.g., perhaps something in the Florida dialog confused BERT), without further analysis we cannot establish causality.

A.6 Model Training, Implementation, and Computation

Models were developed on a single machine with eighty Intel 2.0GHz CPUs, 256GB RAM, and eight Tesla V100 graphics cards. Each model was trained and evaluated on a single graphics cards with hyper-parameter sweeps parallelized across the eight cards. Our best model (CHARM), has 26,970,475 parameters, takes two hours and eighteen minutes to train, and early stops on epoch fifteen.

AllenNLP configuration files and software dependencies (including version) are included with our code at github.com/facebookresearch/curiosity.

A.7 MS Marco Conversational Sample Queries

Conversational MS MARCO is a search dataset that partially inspired this work. Assistant messages should prompt followup queries like in Table A.5.

Topic: <u>Djibouti</u> Aspects: <i>Economy, Politics</i> Known Entities: France, Africa	
Dialog Acts	Messages
U request topic	Hi can you tell me about Djibouti?
A inform response	The country of Djibouti is officially called the Republic of Djibouti is located in the horn of Africa. It was created out of French Somaliland. It is also recorded the country was colonized by France in 1800s and later gained its independence on June 27, 1977.
U request aspect	Can you tell me about their Politics
A inform response	After its independence, The country had two regiments commanded by the French officers. The country also maintains its close ties with Somalia, Ethiopia, France and the United States.
U request aspect	Any other information about the French and their impact of the politics?
A inform response	There are no direct information on its effect on the country's politics but even though the country is already independent. The 13th Demi-brigade of the French Foreign Legion is still stationed in this country. It is also the largest french military presence abroad. commanded by a 3-star general.
U request aspect	Heavy French influence. what other interesting information on politics do you have?
A inform response	In recent years, the country has improved its training techniques, military command, and information structure. They also became more self-reliant in supplying their military and assist in peace-keeping missions and provide military assistance to countries that officially ask them for help.
U request aspect, feedback positive	Wow Interesting, Can you now tell me about the economy they have?
A offer aspect	Sorry, I do not have that information. But I can tell you more about Politics.
U offer accept	Sure
A inform response	Camp Lemonnier is rented by the United State as a Naval Expeditionary Based for \$63 Million a year. While Japan and France each pays \$30 million a year and China pays \$20 Million a year.
U request other	Lastly, any fun facts?
A inform response	I am not sure if this is a fun fact, but the country's gross domestic product expanded by more than 6%. From \$341 Million to 1.5 Billion
U	That's a huge increase. thank you for all your help
A	You are welcome

Table A.1: Example dialog #1 from Curiosity. (U: User, A: Assistant)

Topic: <u>British Columbia</u> Aspects: <i>Government and politics, Culture</i> Known Entities: Canada, Seattle	
Dialog Acts	Messages
U request topic	Hi! Can you help me learn some basic information about British Columbia? I don't know much except that it's located in Canada.
A inform response	Yes, British Columbia is the westernmost province of Canada and is located between the Rocky Mountains and the Pacific Ocean.
U request aspect, feedback positive	I didn't know it was on the coast! What can you tell me about government and politics there?
A inform response	One interesting fact about the government is that the Green Party plays a larger role in this province than it does in other provinces of Canada.
U request followup, feedback positive	Interesting. What can else you tell me about the Green Party?
A inform response	The New Democratic Party and the Green Party caucuses together control 44 seats. Which seems like a lot but the British Columbia Green Party only takes up 3 of those 44 seats.
U request aspect	That's a pretty small influence. Can you tell me some fun culture facts about British Columbia?
A	I am sorry I do not have any information on their culture right now.
U request topic	That's okay. What other fun facts can you share?
A inform response	Interestingly, Queen Victoria chose British Columbia to distinguish what was the British sector of the Columbia District from the United States which became the Oregon Territory on August 8, 1848.
U request aspect	So that's why it has "British" specifically as part of it's name! Makes sense. Are there any sports or outdoor activities that are popular in British Columbia?
A inform response	Horseback riding is enjoyed by many British Columbians.
U	Thanks for your help today. Now I know more than I did before.
A	No problem, it was a pleasure.

Table A.2: Example dialog #2 from Curiosity. (U: User, A: Assistant). After mentioning the Green Party, the user asks a specific followup question; we use these interactions to estimate implicit preference.

Label: paraphrase-correct

Message: I do not have the information available but I can tell you 10 million tonnes were exported annually for, Cardiff dock in the years leading up to the First World War.

Fact: In the years leading up to the First World War , more than 10 million tonnes of coal was exported annually from Cardiff Docks .

Label: copy

Message: Media and journalists in Namibia are represented by the Namibia chapter of the Media Institute of 'southern Africa and the Editors Forum of Namibia.

Fact: Media and journalists in Namibia are represented by the Namibian chapter of the Media Institute of Southern Africa and the Editors ' Forum of Namibia .

Label: paraphrase-multiple

Message: The film industry has a rich history in Indonesian culture and the Indonesian Film Festival has been held regularly since 1955.

Fact: Indonesia has held annual film festivals and awards , including the Indonesian Film Festival (Festival Film Indonesia) that has been held intermittently since 1955 .

Label: context

Message: EU means European Union. GDP per capital ranges from 188% of EU average in Bratislava to 54% in eastern Slovakia

Fact: GDP per capita ranges from 188 % of EU average in Bratislava to 54 % in Eastern Slovakia .

Label: paraphrase-correct

Message: Sudan was considered the 17th growing economy in the world. Its faster development of the country from oil profits.

Fact: Sudan , officially the Republic of the Sudan , is a country in Northeast Africa .

Label: paraphrase-correct

Message: There is a mountain range called the Annamite Range that runs along eastern border to Vietnam and another range, named Luang Prabang Range, near the Thailand border.

Fact: The Mekong River forms a large part of the western boundary with Thailand , where the mountains of the Annamite Range form most of the eastern border with Vietnam and the Luang Prabang Range the northwestern border with the Thai highlands .

Label: copy

Message: In 1603, the kingdoms of England, Scotland and Ireland were united in a personal union when James VI, king of Scots, inherited the crowns of England and Ireland and moved his court from Edinburgh to London; each country nevertheless remained a separate political entity and retained its separate political, legal and religious institutions. Would you like to know anything else?

Fact: In 1603 , the kingdoms of England , Scotland and Ireland were united in a personal union when James VI , King of Scots , inherited the crowns of England and Ireland and moved his court from Edinburgh to London ; each country nevertheless remained a separate political entity and retained its separate political , legal , and religious institutions .

Label: copy

Message: The A31 connects to the A35 at Bere Regis , and passes east through Wimborne and Ferndown to Hampshire where it later becomes the M27

Fact: The A31 connects to the A35 at Bere Regis , and passes east through Wimborne and Ferndown to Hampshire , where it later becomes the M27 .

Label: context

Message: I'm not sure of that. One fact I do have is that by 1857, Chicago was Illinois' largest city.

Fact: By 1857 , Chicago was Illinois 's largest city .

Label: paraphrase-correct

Message: On March 2, 2008, Dmitry Medvedev was elected President of Russia and Vladimir Putin became Prime Minister.

Fact: On 2 March 2008 , Dmitry Medvedev was elected President of Russia while Putin became Prime Minister .

Liked	Correct Model	Message
No	BERT	You are welcome!
Yes	BERT	I'm sorry I don't have anymore information about the etymology of Tunisia, but what I can tell you is that Tunisia Sports City is a whole sports city being constructed in Tunis
Yes	BERT	Yes Buddhism is a dominant influence in Lao culture. It has been great helping you.
Yes	CHARM	Florida is a state in the southeast United States. What would you like to know?
Yes	BERT	They have an average daily temperature of 70.7, it's the warmest state in the U. S.
Yes	CHARM	Yes, I can. Florida is nicknamed the "Sunshine State", but severe weather is a common occurrence.
Yes	BERT	Hello, Indonesia is part of the Malay Islands and is in Southeast Asia. Would you like to know more about the history?
Yes	BERT	I do not have etymologic information, would you like to know more about the economy? I can tell you thank Indonesia develops military and commuter aircraft.

Table A.4: To compare like prediction between models, we randomly sample thirty dialogs and obtain predictions from CHARM and BERT. The table only shows messages where the model predictions disagree and indicates which model was correct. Dialogs are delineated by horizontal lines. Unfortunately, from only these examples we cannot determine why the CHARM model errors in most of these predictions.

Query

What is a physician's assistant?
What are the educational requirements required to become a physician's assistant?
What does the education to become a physician's assistant cost?
What's the average starting salary of a physician's assistant in the UK?
What's the average starting salary of a physician's assistant in the US?
What school subjects are needed to become a registered nurse?
What is the physician's assistant average salary vs a registered nurse?
What the difference between a physician's assistant and a nurse practitioner?
Do nurse practitioners or physician's assistant's make more?
Is a physician's assistant above a nurse practitioner?
What is the fastest way to become a nurse practitioner?
How much longer does it take to become a doctor after being a nurse practitioner?
What are the main breeds of goat?
Tell me about boer goats.
What goat breed is good for meat?
Are angora goats good for meat?
Are boer goats good for meat?
What are pygmy goats used for?
What goat breed is the best for fiber production?
How long do Angora goats live?
Can you milk Angora goats?
How many Angora goats can you have per acre?
Are Angora goats profitable?

Table A.5: An exemplar query chain from the conversational variant of MS MARCO. An ideal assistant should answer these questions and inspire these types of followup questions.

Appendix B: Leaderboard

This appendix supports Chapter 4 by showing additional SQuAD examples, feature descriptions, reproducibility details, minor results related to the work’s main experiments, and discussions of multidimensional clustering and statistical testing.

B.1 SQuAD Item Examples

On our project page at irt.pedro.ai, we include annotations from Figure 4.10 in human and machine-readable formats and provide an interactive web interface to inspect the parameters of the IRT-base, IRT-disc, and IRT-feas models. Figure B.4 shows the feasibility distribution corresponding to Figure 4.1.

B.2 Logistic Regression Features

The linear model (§4.4.2) includes features based on item IDs, subject IDs, textual features of the question, context, and answer, and topic model features. Table B.1 lists the feature names from Figure 4.4 with descriptions of each. When IRT features or the statistics features are used, they include interaction terms with themselves.

B.3 IRT Model Type Correlation

Although each IRT model differs in expressiveness, they should—in general—produce similar results. This is confirmed by computing the Kendall’s rank correlation between the subject abilities and item difficulties (Table B.2).

B.4 Ranking Stability Experiments

Here we provide further details for the ranking stability experiments (§4.4.2.3). First, we filter from the 161 subjects that have development set scores to the 115 that also have test set scores.¹ In our simulation, we run 10 trials for every sample size; sample size begin at 100 and advances by 100. In addition to these, we also run for sample sizes 25, 50, and 75. Since each sample can be no larger than half the dataset, we stop at half the dataset.

¹The SQuAD organizers curate the test set subjects to avoid overfit, garbage, or duplicate submissions.

Discriminability: -9.63 **Difficulty:** -0.479 **Feasibility:** 0.614 **Mean Exact Match:** 0.472

Wikipedia **Page:** Economic inequality **Question** **ID:**
572a1c943f37b319004786e3

Question: Why did the demand for rentals decrease?

Official Answer: demand for higher quality housing

Context: A number of researchers (David Rodda, Jacob Vigdor, and Janna Matlack), argue that a shortage of affordable housing – at least in the US – is caused in part by income inequality. David Rodda noted that from 1984 and 1991, the number of quality rental units decreased as the demand for higher quality housing increased (Rhoda 1994:148). Through gentrification of older neighbourhoods, for example, in East New York, rental prices increased rapidly as landlords found new residents willing to pay higher market rate for housing and left lower income families without rental units. The ad valorem property tax policy combined with rising prices made it difficult or impossible for low income residents to keep pace.

Figure B.1: The example from SQuAD with the lowest discriminability. Surprisingly, it had a *negative* discriminability, implying that the less skilled a subject is, the more likely their response is to be correct.

B.4.1 Development and Test Set Correlations

Table B.3 uses a IRT-disc model since we noticed that in comparison IRT-feas overfit the data, yielding worse results. The correlations with the full data are all strong, but not the same. Taking all these results, we conclude that—at least on SQuAD—IRT rankings are modestly more reliable than classical rankings.

B.5 The IRT Statistical Test

The IRT test differs in two substantial ways from other tests: (1) it does not assume that items are equally informative and (2) it does assume that the informativeness of items is a function of the subject’s skill θ_j . In the literature, this is closely connected to *reliability* (Tague-Sutcliffe, 1992) and each item provides information about the location of θ_j ; as we accumulate more evidence for the location of θ_j the error of estimation decreases. It is a well known result in IRT that standard error of estimate (SEE) $\sigma(\hat{\theta}|\theta)$ varies with respect to the agent location parameter θ (De Ayala, 2013, p. 30) and is connected to the Fisher information

$$I_i(\theta) = \frac{(p'_i)^2}{p_i(1 - p_i)} \tag{B.1}$$

of each item. For a 2PL model, information

$$I_i(\theta) = \gamma^2 p_i(1 - p_i) \tag{B.2}$$

Discriminability: 2.1	Difficulty: 2.38	Feasibility: 0.995	Mean Exact Match: 0.00621	Mean F1: 0.546
Wikipedia Page: 57268f2bf1498d1400e8e3c4	Question ID: European Union Law			
Question: What reform was attempted following the Nice Treaty?				
Official Answer: an attempt to reform the constitutional law of the European Union and make it more transparent				
Context: Following the Nice Treaty, there was an attempt to reform the constitutional law of the European Union and make it more transparent; this would have also produced a single constitutional document. However, as a result of the referendum in France and the referendum in the Netherlands, the 2004 Treaty establishing a Constitution for Europe never came into force. Instead, the Lisbon Treaty was enacted. Its substance was very similar to the proposed constitutional treaty, but it was formally an amending treaty, and – though it significantly altered the existing treaties – it did not completely replace them.				

Figure B.2: This example shows that the answer span is likely too large, causing models to fail in both SQuAD’s exact match and F1 metrics.

is maximized when $p_i = (1 - p_i)$. Since Fisher information is additive, the information of the evaluation set is maximal when items have a 50% chance of being responded to correctly. As derived by De Ayala (2013, p. 102), the standard error of estimation

$$SEE(\theta) = \sqrt{\frac{1}{\sum_i I_i(\theta)}}. \tag{B.3}$$

is computed by accumulating the information gained from each item. Given two subjects X and Y , one can use the probability distribution of score differences

$$N(\theta_Y - \theta_X, SEE(\theta_X)^2 + SEE(\theta_Y)^2) \tag{B.4}$$

to compute the probability that the difference in skill is greater than two standard errors which corresponds to an $\alpha \leq .05$ significance level.

B.6 Multidimensional IRT Clustering

While we achieve strong held-out accuracy with the 10 dimensional IRT (IRT-vec), we had limited success in interpreting parameters. We use TSNE² plots overlaid with features like item accuracy, the question’s Wikipedia page, if the question was answerable, length of questions, and topic model weights. Of these, item accuracy and answerability showed the most obvious patterns (Figure B.5).

²We use openTSNE (Poličar et al., 2019) with default parameters.

Discriminability: 8.01 **Difficulty:** -1.41 **Feasibility:** 0.939 **Mean Exact Match:** 0.64 **Mean F1:** 0.667
Wikipedia Page: Normas **Question ID:** 56de10b44396321400ee2595
Question: Who did the Normans team up with in Anatolia?
Official Answer: Turkish forces
Context: Some Normans joined Turkish forces to aid in the destruction of the Armenians vassal-states of Sassoun and Taron in far eastern Anatolia. Later, many took up service with the Armenian state further south in Cilicia and the Taurus Mountains. A Norman named Oursel led a force of "Franks" into the upper Euphrates valley in northern Syria...

Figure B.3: This highly discriminative question succeeds because there are many plausible answers. For example, although only “Turkish forces” is correct, some models answer “the Armenian state.”

Feature	Description
All	All the features
IRT	IRT values for difficulty, discriminability, feasibility, and ability
Item ID	The item’s ID
Subject ID	The subject’s ID
Question	Question words
Context	Context words
Stats	Lengths of question and context; answerability, answer position and length; difficulty from Sugawara et al. (2017)
Subject & Item ID	Item and Subject ID
Topics 1K	Topic weights of question words
Title	Wikipedia page title words
Baseline	No features, majority class baseline

Table B.1: The linear model integrates a variety of features to determine which are most predictive of a subject responding correctly to an item.

B.7 Reproducibility Checklist

Here we provide reproducibility details to complement our source code and data release at <https://irt.pedro.ai>.

B.7.1 Software and Parameters

All reported IRT models are implemented in PyTorch ([Paszke et al., 2019](#)) and Pyro ([Bingham et al., 2018](#)). Linear models are trained with Vowpal Wabbit ([Agarwal et al., 2014](#)). The topic model that generated features for the linear model uses Mallet ([McCallum, 2002](#)).

The IRT models have parameters proportional to the number of subjects m and the number of items n . The IRT-base has one parameter per subject and one

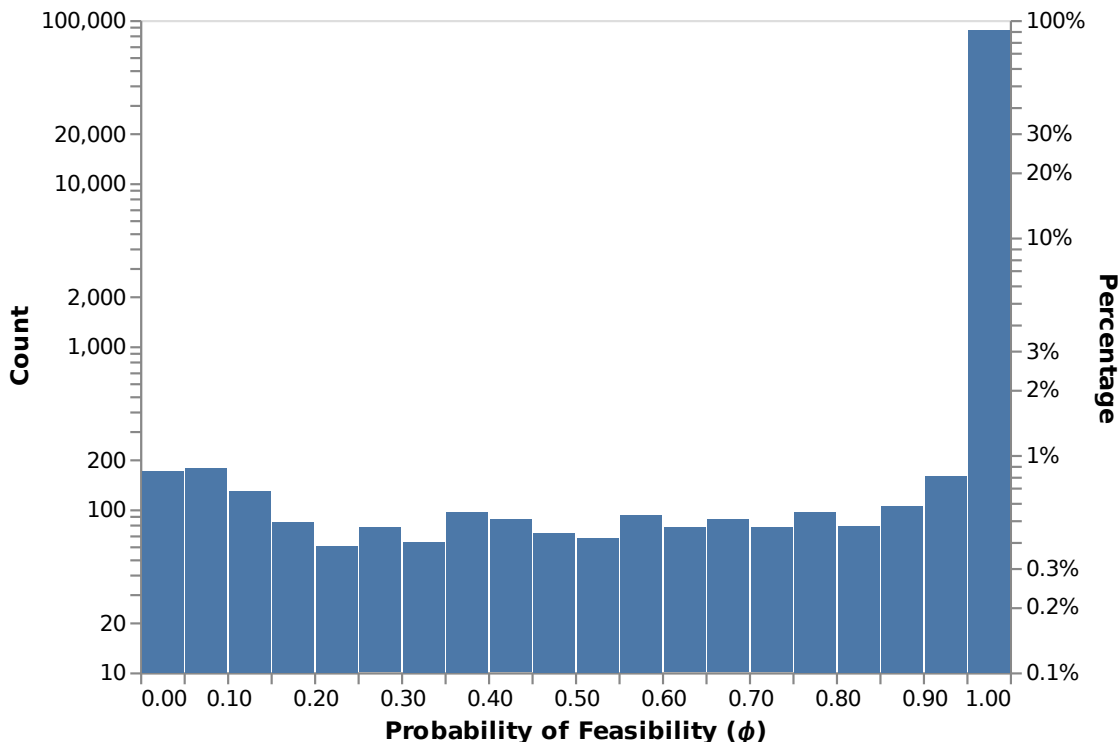


Figure B.4: The feasibility parameter λ of our IRT model represents the probability that an example is unsolvable. For example, annotation error could lead to an example always being scored incorrectly—regardless of how good the model is. In SQuAD 2.0, $\lambda < .434$ in the 5% percentile, $\lambda < .698$ for the 7.5%, and $\lambda < .931$ in the 10% percentile.

parameter per item. The IRT-disc has one parameter per subject and two parameters per item. The IRT-feas has one parameter per subject and three parameters per item. The IRT-vec has ten parameters per subject and thirty parameters per item.

B.7.2 Hyperparameters

We did not invest significant effort on hyper-parameter tuning the IRT models and instead used the defaults in the py-irt software³ provided by Lalor et al. (2019). The IRT-base, IRT-disc, and IRT-feas models were trained for 1000 epochs with no early stopping conditions and a learning rate of 0.1 with ADAM (Kingma and Ba, 2015). The IRT-vec model was trained for 2500 epochs and used 10 dimensions.

In the linear model, we used a Hyperopt-based (Bergstra et al., 2013) tool provided by Vowpal Wabbit⁴ for hyper parameter search. For each LM, the tool spent 20 iterations optimizing the learning rate, L2 regularization, and number of bits against the logistic loss function. The learning rate was searched from .001 to 10 with loguniform sampling, L2 regularization from $1e-8$ to 1, and bits from 20 to

³<https://github.com/jplalor/py-irt>

⁴https://github.com/VowpalWabbit/vowpal_wabbit/blob/master/utl/vw-hyperopt.py

Ability	IRT-feas	IRT-disc	IRT-base
IRT-feas	1.00	0.947	0.895
IRT-disc	0.947	1.00	0.907
IRT-base	0.895	0.907	1.00

Table B.2: Table entries are Kendall’s τ rank correlation of IRT subject ability between rows and columns. Generally, the models agree on the ranking with the IRT-feas and IRT-disc having the strongest correlation.

	EM _{dev}	EM _{test}	Ability _{dev}	Ability _{test}
EM _{dev}	1.00	0.953	0.954	0.931
EM _{test}	0.953	1.00	0.944	0.947
Ability _{dev}	0.954	0.944	1.00	0.950
Ability _{test}	0.931	0.947	0.950	1.00

Table B.3: Entries are Kendall’s rank correlation between rows and columns. Scores are SQuAD Exact Match (EM) and IRT-disc ability.

23 as categorical variables.

The topic model that generated features for the linear model used `mallet`, and we followed the recommendations of the software⁵ to set hyper parameters. Specifically, we used an optimization interval of 10, removed stop words, trained for 1000 iterations, and used a document topic threshold of 0.05. Each document was comprised of the Wikipedia page title and the question text.

B.7.3 Computational Resources

The majority of experiments were conducted on a single workstation with an Intel i7-7700K CPU, 47GB of RAM, and an Nvidia 1080Ti. The average runtime for the IRT-feas model on CPU is 113 seconds with a standard deviation of 2.31 over 5 trials. The average runtime of the IRT-vec model on GPU is 110 seconds with a standard deviation of 0.5 over 5 trials.

Since each ranking stability required (§4.4.3.1) re-training an IRT-feas model on each subset, we parallelized this experiment on a CPU cluster where each trial received two CPU cores and 16GB of RAM. In total, this included 520 trials which corresponds to twice that many trained IRT models since one model is trained on each subset of the data.

⁵<http://mallet.cs.umass.edu/topics.php>

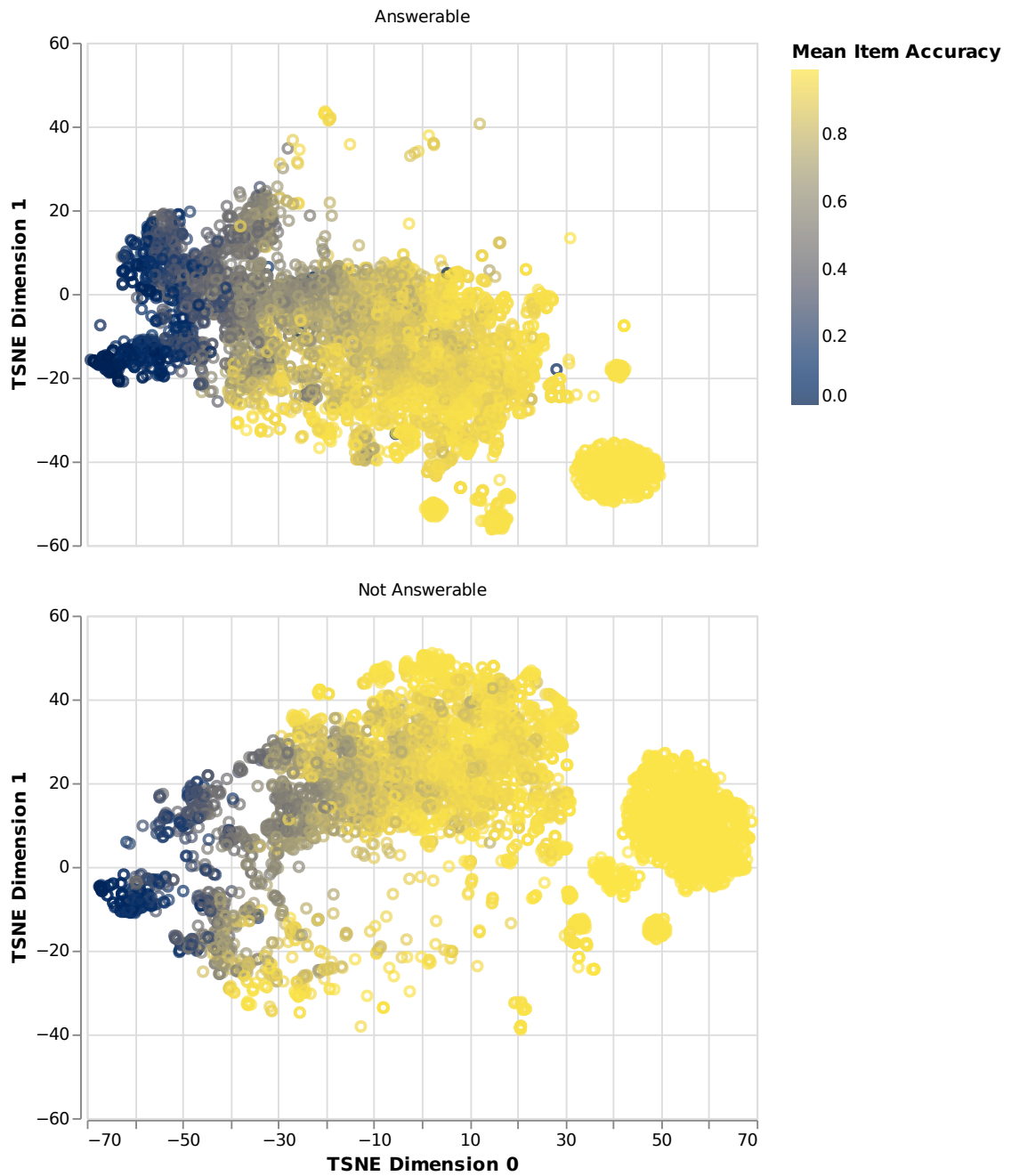


Figure B.5: In SQuAD, tSNE shows a relationship between mean exact match (item accuracy) and answerability with respect to multidimensional difficulty and discriminability.

Appendix C: Quizbowl

This appendix provides additional details on preprocessing and model features in Chapter 5.

C.1 Preprocessing

This section provides a detailed description of preprocessing on the question dataset.

C.1.1 Aligning and De-duplicating Questions

Since we obtain machine-readable versions of questions from two online sources it is necessary to ensure that we do not include the same question twice. We use the metadata associated with each question such as tournament and year. As part of our preprocessing we manually align the values of these fields.¹ We use these fields to ensure that questions for each tournament and year are included only once.

C.1.2 Textual Preprocessing

Models should define their own textual preprocessing so we only preprocess the text to remove QB specific artifacts. Most of these artifacts are instructions to the moderator or organizer such as “MODERATOR NOTE:”, “Description Required”, “15 pts:”, or a reference to the category of the question; we use regular expression rules to remove these. Since we report results on accuracy after the first sentence in questions we also provide a set of canonical sentence tokenization indices computed using spacy.²

C.1.3 Fold Assignment

We divide QANTA dataset questions into training, development, and test folds based on the competitiveness and year of the source tournament. Since championship tournaments typically have the highest quality questions we use questions from championship tournaments 2015 and onward as development and test sets. All other questions are used as the training set.

¹We also align category and sub-category fields.

²<https://spacy.io>

Table 5.3 shows the divisions of each fold; each train, dev, or test fold is assigned to be used for either determining what to answer (guessing) or when to answer (buzzing). Questions in “guess” folds are used for developing question answering systems as in Section 5.5. Questions in the “buzz” folds are used for developing agents that decide when to answer as in Section 5.6.

When we assign folds to QB questions we aim to create useful splits for guessing and buzzing while preserving the integrity of the development and test sets. Namely, when we create test and development folds we make the division into folds not depend on whether or not gameplay data exists. If it were the case that by making this unconditional assignment the number of questions with gameplay data is too small this would be a problem. We do not find this to be a problem however.

For test set questions this is easily accomplished by using an implicit quality filter and a temporal split; use only questions from national championship tournaments, questions from 2016 are used in the buzzing test set, and questions from 2017 and 2018 are used for the guessing test set. Following this we pair the test fold for buzzing with gameplay data, and are fortunate that the number of questions is not small.

To create the development sets we use questions from 2015 which are randomly split with equal probability into guessing and buzzing specific folds. Similarly to the test set we associate gameplay data after this assignment occurs to preserve its integrity against any bias that conditioning on having gameplay data would have.

For the training data we make a weaker attempt to eliminate bias in favor of ensuring that the training folds for guessing and buzzing are large enough. We first divide the training questions with an 80/20 split. Questions in the eighty percent split are assigned to the guessing fold. Each remaining question is assigned to the buzzing fold if it has gameplay data, otherwise it is assigned to the guessing fold. Figure 5.3 shows the result of this folding procedure.

C.1.4 Matching QB Answers to Wikipedia Pages

The automatic rule based part of this process is composed of two phases: an expansion phase in that produces variants of the answer text, and a match phase that determines when one of these variants is a match to a Wikipedia page. The rules in the expansion phase can be as simple as exact text match to expanding “The {Master of Flémalle} or Robert {Campin}” to “{Master of Flémalle}” and “Robert {Campin}”. In this case, multiple matches result in “Robert Campin” being the answer page: after removing braces “The Master of Flémalle” Wikipedia redirects to “Robert Campin” and “Robert Campin” is also an exact match. These rules are incredibly effective at finding answers buried in QB specific notation such as the random sample in Table C.1. When matches disagree we use the match that modified the original answer text the least.

There are inevitably cases where the automatic system fails to find a match, or finds the wrong match. Qualitatively these are often caused by disambiguation errors such as failing to differentiate between “Guernica” the city versus the painting by Picasso, small differences in answer strings, and when there is no suitable Wikipedia

Original QB Answer	Matched Wikipedia Page
Nora Helmer	A_Doll's_House
{Gauss}'s law for the electric field	No Mapping Found
Thomas Hutchinson	Thomas_Hutchinson_(governor)
linearity	Linearity
{caldera}s	Caldera
William Holman {Hunt}	William_Holman_Hunt
{plasma}s	Plasma_(physics)
{Second Vatican Council} [or {Vatican II}]	Second_Vatican_Council
{Jainism}	Jainism
{Electronegativity}	Electronegativity
Hubert Selby, Jr.	Hubert_Selby_Jr.
(The) Entry of Christ into Brussels (accept equivalents due to translation)	Christ's_Entry_Into_Brussels_in_1889
Depictions of Speech [accept equivalents]	No Mapping Found
stress	Stress_(mechanics)

Table C.1: A random sample of QB answer strings and their matched Wikipedia pages. Answer mappings are easy to obtain accurately since most failures in exact matching are due to QB specific syntax that can be accounted for by rule based matching. Combined with manual annotation to find common non-exact matches, this process succeeds on 119,093 of 132,849.

page. To correct or verify these errors we (the authors), and skilled members of the QB community (such as tournament organizers and participants from our exhibition matches) manually annotated a significant fraction of the training data, and all the test data.

Rather than doing manual annotation of each question, we begin by defining mappings of answer strings to Wikipedia pages so that when that string occurs multiple times it does not require manual annotation for every occurrence of that answer in questions. However, this has the serious drawback that if the answer string is ambiguous then it may result in mislabeled answers. To avoid this problem we design a manual process whereby annotators update three sets of answer-to-Wikipedia mappings: unambiguous, ambiguous, and direct mappings.

Unambiguous annotations contain a list of answer strings that when seen map to a specific Wikipedia page. As the name implies, we only insert annotations here when the answer unambiguously identifies the corresponding Wikipedia page. Ambiguous annotations similarly contain a list of answer strings, but are paired with

a list of disambiguation words. If the answer string is seen, at least one word is in the question text, and there are no other ambiguous matches, then it is mapped. For example, if the answer string is “amazon” and the question contains the word “river” then we assume “Amazon river” is the correct page while if the question mentions “bezos” then the correct page is “Amazon (company)”. Finally, direct mappings match the answer for specific questions.

The last major design decision in this process addresses how we prevent information from the test data to leak into the training data. The root of the data leak issue is that the distribution of answers between training and test data often results in only approximately 80% of test set answers occurring in the training data. We observed this phenomena empirically in both our data and the distribution of answers from our numerous exhibition events. If all answer strings are naively combined, then mapped, this implies that the training data will be biased towards its answers containing an over abundance of test set answers. A major difference between this and prior versions of the QANTA dataset is finding and fixing this issue.

We correct this error by separating the answer string pool for training and test questions. Although this results in more annotation work, it avoids information leakage. While reviewing our annotation procedure we noticed another source of bias. Recall that we do not exhaustively annotate the training data. In our initial annotation we did not fully annotate the test data, and by doing so introduced a bias towards easier-to-annotate questions in the test set. To eliminate this bias—and make it as similar to playing a QB tournament as possible—we annotated every question in the test set.³

C.1.5 Buzzer features

The guesser updates its list of guesses whenever a new word of the QB question is revealed. At each time step, the buzzer extracts features from both the current and all past guesses and predict whether the current guess is correct. It is important to include past guesses as the dynamics of the guesser’s confidence contains strong signal about its correctness: the guesser usually starts with some random guess when little information is provided, then fluctuates between several plausible answers—just as humans do, and finally stabilizes to a single answer, at which point the buzzer should buzz. Below is the full list of buzzer features we use in the experiments:

- Probabilities of the current top 3 guesses
- Change of top 3 probabilities from the previous step
- Gaps between probabilities of the top 3 guesses
- Binary indicator of whether each of the top 5 guesses increased its ranking from previous step

³Specifically, we either pair each test set answer strings with a Wikipedia title or mark it as not having a corresponding Wikipedia title.

Category	N	Percent
Pop Culture	55	40%
History	26	19%
Science	20	15%
Other	13	9.6%
Social Science	7	5.1%
Geography	6	4.4%
Religion	2	1.5%
Literature	5	3.7%
Philosophy	1	0.74%
Fine Arts	1	0.74%
Total w/ Category	136	100%
No Category	14	
Total	150	

Table C.2: A breakdown of NaturalQuestion example topics using QB categories. Most questions are about pop culture and the distribution includes many fewer questions about Literature and Fine Arts.

- Mean and variance of probabilities of the current top 3 guesses
- Mean and variance of probabilities of the previous top 3 guesses

C.2 Natural Questions Categories

Section 5.3.2 analyzes the topical diversity of QB questions and makes comparisons to NaturalQuestions. To compare to NaturalQuestions—which does not have category labels—we annotated a random subset of 150 questions using QB’s categories (Table C.2).

Appendix D: Centaur Authorship of Quizbowl Questions

This appendix contains an analysis of automatically created adversarial examples and details of the Studio OUSIA model.

D.1 Failure of Syntactically Controlled Paraphrase Networks

Sentence	Success/Failure Phenomena
its types include “frictional”, “cyclical”, and “structural” its types include “frictional”, and structural	Missing Information ✗
german author of the sorrows of young werther and a two-part faust german author of the sorrows of mr. werther	Lost Named Entity ✗
name this elegy on the death of john keats composed by percy shelley name was this elegy on the death of percy shelley	Incorrect Clue ✗
identify this play about willy loman written by arthur miller so you can identify this work of mr. miller	Unsuited Syntax Template ✗
he employed marco polo and his father as ambassadors he hired marco polo and his father as ambassadors	Verb Synonym ✓

Table D.1: Failure and success cases for SCPN. The model fails to create a valid paraphrase of the sentence for 97% of questions.

We apply the Syntactically Controlled Paraphrase Network (Iyyer et al., 2018, SCPN) to QB questions. The model operates on the sentence level and cannot paraphrase paragraphs. We thus feed in each sentence independently, ignoring possible breaks in coreference. The model does not correctly paraphrase most of the complex sentences present in QB questions. The paraphrases were rife with issues: ungrammatical, repetitive, or missing information.

To simplify the setting, we focus on paraphrasing the shortest sentence from each question (often the final clue). The model still fails in this case. We analyze a random sample of 200 paraphrases: only six maintained all of the original information.

Table D.1 shows common failure cases. One recurring issue is an inability to maintain the correct named entities after paraphrasing. In QB, maintaining entity information is vital for ensuring question validity. We were surprised by this failure

because SCPN incorporates a copy mechanism.

D.2 Studio Ousia QB Model

The Studio Ousia system works by aggregating scores from both a neural text classification model and an IR system. Additionally, it scores answers based on their match with the correct entity type (e.g., religious leader, government agency, etc.) predicted by a neural entity type classifier. The Studio Ousia system also uses data beyond QB questions and the text of Wikipedia pages, integrating entities from a knowledge graph and customized word vectors ([Yamada et al., 2018b](#)).

Bibliography

2020. Canon - qb wiki. <https://www.qbwiki.com/wiki/Canon>. Accessed: 2021-03-05.
- Firas Abuzaid, Geet Sethi, Peter Bailis, and Matei Zaharia. 2019. To index or not to index: Optimizing exact maximum inner product search. In 2019 IEEE 35th International Conference on Data Engineering (ICDE), pages 1250–1261. IEEE.
- Alekh Agarwal, Olivier Chapelle, Miroslav Dudík, and John Langford. 2014. A reliable effective terascale linear learning system. Journal of Machine Learning Research, 15:1111–1133.
- Luis von Ahn. 2006. Games with a purpose. Computer, 39:92 – 94.
- Luis von Ahn and Laura Dabbish. 2004. Labeling images with a computer game. In International Conference on Human Factors in Computing Systems.
- Luis von Ahn and Laura Dabbish. 2008. Designing games with a purpose. Communications of the ACM, 51(8):58–67.
- Xavier Amatriain and Justin Basilico. 2012. Netflix recommendations: Beyond the 5 stars (part 1). <https://netflixtechblog.com/netflix-recommendations-beyond-the-5-stars-part-1-55838468f429>. Accessed: 2021-03-15.
- National Council on Measurement in Education, Joint Committee on Standards for Educational and Psychological Testing (U.S.) American Educational Research Association, American Psychological Association. 2014. Standards for educational and psychological testing. American Educational Research Association, Washington, DC.
- Saleema Amershi, Max Chickering, Steven M Drucker, Bongshin Lee, Patrice Simard, and Jina Suh. 2015. ModelTracker: Redesigning performance analysis tools for machine learning. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI '15, pages 337–346, New York, NY, USA. Association for Computing Machinery.

- Christine M Anderson-Cook, Kary L Myers, Lu Lu, Michael L Fugate, Kevin R Quinlan, and Norma Pawley. 2019. How to host an effective data competition: Statistical advice for competition design and analysis. Statistical Analysis and Data Mining: The ASA Data Science Journal, 12(4):271–289.
- Dustin Arendt, Zhuanyi Huang, Prasha Shrestha, Ellyn Ayton, Maria Glenski, and Svitlana Volkova. 2021. CrossCheck: Rapid, reproducible, and interpretable model evaluation. In Proceedings of the Second Workshop on Data Science with Human in the Loop: Language Advances. Association for Computational Linguistics.
- Sylvain Arlot and Alain Celisse. 2010. A survey of cross-validation procedures for model selection. Statistics surveys, 4:40–79.
- Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. Computational Linguistics, 34(4):555–596.
- Richard D Arvey, Thomas J Bouchard, John B Carroll, Raymond B Cattell, David B Cohen, Rene V Dawis, and L Willerman. 1994. Mainstream science on intelligence. Wall Street Journal, 13(1):18–25.
- David Baber. 2015. Television Game Show Hosts: Biographies of 32 Stars. McFarland.
- Ricardo Baeza-Yates, Aristides Gionis, Flavio Junqueira, Vanessa Murdock, Vasilis Plachouras, and Fabrizio Silvestri. 2007. The impact of caching on search engines. In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '07, pages 183–190, New York, NY, USA. Association for Computing Machinery.
- Frank B Baker. 2001. The Basics of Item Response Theory. ERIC.
- Solon Barocas, Moritz Hardt, and Arvind Narayanan. 2019. Fairness and Machine Learning. fairmlbook.org. <http://www.fairmlbook.org>.
- Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. 2020. Beat the AI: Investigating Adversarial Human Annotation for Reading Comprehension. Transactions of the Association for Computational Linguistics, 8:662–678.
- Hannah Bast, Florian Baurle, Björn Buchhold, and Elmar Haussmann. 2014. Easy access to the freebase dataset. In Proceedings of the World Wide Web Conference.
- Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In Proceedings of the International Conference on Learning Representations.

- Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. In Transactions of the Association for Computational Linguistics, pages 49–72.
- Nicholas J Belkin, Colleen Cool, Adelheit Stein, and Ulrich Thiel. 1995. Cases, scripts, and information-seeking strategies: On the design of interactive information retrieval systems. Expert systems with applications, 9(3):379–395.
- Emily M Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. Transactions of the Association for Computational Linguistics, 6:587–604.
- Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In Proceedings of the ACM Conference on Fairness, Accountability, and Transparency.
- Emily M Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- R. Benjamin. 2019. Race After Technology: Abolitionist Tools for the New Jim Code. Polity Press.
- James Bennett and Stan Lanning. 2007. The netflix prize. In Proceedings of KDD cup and workshop.
- Jonathan Berant, Andrew Chou, Roy Frostig, and Percy S. Liang. 2013. Semantic parsing on freebase from question-answer pairs. In Proceedings of Empirical Methods in Natural Language Processing.
- James Bergstra, Daniel Yamins, and David Cox. 2013. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In Proceedings of the 30th International Conference on Machine Learning, volume 28 of Proceedings of Machine Learning Research, pages 115–123. PMLR.
- D E Berlyne. 1954. A theory of human curiosity. British journal of psychology, 45(3):180–191.
- Darse Billings, Denis Papp, Jonathan Schaeffer, and Duane Szafron. 1998. Opponent modeling in poker. In Association for the Advancement of Artificial Intelligence.
- Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. 2018. Pyro: Deep Universal Probabilistic Programming. Journal of Machine Learning Research.

- Abeba Birhane and Vinay Uday Prabhu. 2021. Large image datasets: A pyrrhic win for computer vision? In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pages 1537–1547.
- Edwin Black. 2012. War Against the Weak: Eugenics and America’s Campaign to Create a Master Race, Expanded Edition. Dialog Press.
- David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. Journal of Machine Learning Research, 3:993–1022.
- Su Lin Blodgett, Solon Barocas, Hal Daumé, III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of “bias” in NLP. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 5454–5476, Online. Association for Computational Linguistics.
- Su Lin Blodgett, Lisa Green, and Brendan O’Connor. 2016. Demographic dialectal variation in social media: A case study of African-American English. In Proceedings of Empirical Methods in Natural Language Processing.
- Avrim Blum and Moritz Hardt. 2015. The ladder: A reliable leaderboard for machine learning competitions. In Proceedings of the International Conference of Machine Learning. PMLR.
- Daniel G Bobrow. 1964. Natural language input for a computer problem solving system. Technical report.
- Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. pages 1–44.
- Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. arXiv preprint arXiv:1506.02075.
- C. Boyce-Jacino and S. DeDeo. 2018. Opacity, Obscurity, and the Geometry of Question-Asking. ArXiv e-prints.
- Jordan Boyd-Graber and Benjamin Börschinger. 2020. What question answering can learn from trivia nerds. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Jordan Boyd-Graber, Shi Feng, and Pedro Rodriguez. 2018. Human-Computer Question Answering: The Case for Quizbowl. Springer.
- Jordan Boyd-Graber, Pedro Rodriguez, Nathan Murphy, and R. Hentzel. 2017. Qanta vs. quiz bowl veterans.

- Jordan Boyd-Graber, Brianna Satinoff, He He, and Hal Daumé III. 2012. Besting the quiz master: Crowdsourcing incremental classification games. In Proceedings of Empirical Methods in Natural Language Processing.
- Laura Briggs. 2003. Reproducing Empire: Race, Sex, Science, and U.S. Imperialism in Puerto Rico. American Crossroads. University of California Press.
- Carla E Brodley and Mark A Friedl. 1999. Identifying mislabeled training data. The journal of artificial intelligence research, 11(1):131–167.
- Noam Brown and Tuomas Sandholm. 2019. Superhuman AI for multiplayer poker. Science, 365(6456):885–890.
- Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Wojciech Gajewski, Andrea Gesmundo, Neil Houlsby, and Wei Wang. 2018. Ask the right questions: Active question reformulation with reinforcement learning. In Proceedings of the International Conference on Learning Representations.
- Chris Buckley and Ellen M Voorhees. 2000. Evaluating evaluation measure stability. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval.
- Harry Bunt, Jan Alexandersson, Jean Carletta, Jae-Woong Choe, Alex Chengyu Fang, Kôiti Hasida, Kiyong Lee, Volha Petukhova, Andrei Popescu-Belis, Laurent Romary, Claudia Soria, and David R. Traum. 2010. Towards an ISO standard for dialogue act annotation. In Proceedings of the Language Resources and Evaluation Conference.
- Harry Bunt, Jan Alexandersson, Jae-Woong Choe, Alex Chengyu Fang, Kôiti Hasida, Volha Petukhova, Andrei Popescu-Belis, and David R. Traum. 2012. ISO 24617-2: A semantically-based standard for dialogue annotation. In Proceedings of the Language Resources and Evaluation Conference.
- Joy Buolamwini and Timnit Gebru. 2018. Gender shades: Intersectional accuracy disparities in commercial gender classification. In Proceedings of the 1st Conference on Fairness, Accountability and Transparency, volume 81 of Proceedings of Machine Learning Research, pages 77–91, New York, NY, USA. PMLR.
- Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the role of Bleu in machine translation research. In Proceedings of the European Chapter of the Association for Computational Linguistics, Trento, Italy. Association for Computational Linguistics.
- B. Barla Cambazoglu, Mark Sanderson, Falk Scholer, and Bruce Croft. 2020. A review of public datasets in question answering research. ACM SIGIR Forum, 54(2).

- Yang Trista Cao and Hal Daumé, III. 2020. Toward Gender-Inclusive coreference resolution. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 4568–4595, Online. Association for Computational Linguistics.
- L R Caporael. 1986. Anthropomorphism and mechanomorphism: Two faces of the human machine. Computers in human behavior, 2(3):215–234.
- Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. 2020. With little power comes great responsibility. In Proceedings of Empirical Methods in Natural Language Processing.
- Dallas Card, Chenhao Tan, and Noah A Smith. 2018. Neural models for documents with metadata. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2020. Extracting training data from large language models.
- Nicholas Carlini and David Wagner. 2017. Adversarial examples are not easily detected: Bypassing ten detection methods. In Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISEC '17, pages 3–14, New York, NY, USA. Association for Computing Machinery.
- David Carmel and Elad Yom-Tov. 2010. Estimating the query difficulty for information retrieval. Synthesis Lectures on Information Concepts, Retrieval, and Services, 2(1):1–89.
- Nicky Case. 2018. How To Become A Centaur. Journal of Design and Science.
- J Mckeen Cattell. 1915. Families of american men of science: Origin, heredity and performance'. Popular Science Monthly, May, pages 248–262.
- Xiaoyong Chai, Lin Deng, Qiang Yang, and Charles X. Ling. 2004. Test-cost sensitive naive bayes classification. Fourth IEEE International Conference on Data Mining (ICDM'04), pages 51–58.
- Seth Chaiklin. 2003. The Zone of Proximal Development in Vygotsky's Analysis of Learning and Instruction, Learning in Doing: Social, Cognitive and Computational Perspectives, page 39–64. Cambridge University Press.
- Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Evaluating question answering evaluation. In Proceedings of the 2nd Workshop on Machine Reading for Question Answering, pages 119–124, Stroudsburg, PA, USA. Association for Computational Linguistics.

- Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2020. MOCHA: A dataset for training and evaluating generative reading comprehension metrics. In Proceedings of Empirical Methods in Natural Language Processing.
- Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the CNN/Daily Mail reading comprehension task. In Proceedings of the Association for Computational Linguistics.
- Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In Proceedings of the Association for Computational Linguistics.
- Danqi Chen and Wen-Tau Yih. 2020. Open-Domain question answering. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts.
- Xilun Chen and Claire Cardie. 2018. Multinomial adversarial networks for multi-domain text classification. In Conference of the North American Chapter of the Association for Computational Linguistics.
- Xiao Cheng and Dan Roth. 2013. Relational inference for wikification. In Proceedings of Empirical Methods in Natural Language Processing.
- Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machine translation: Encoder-decoder approaches. In Proceedings of Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation.
- Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Proceedings of Empirical Methods in Natural Language Processing.
- Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. In Proceedings of Empirical Methods in Natural Language Processing.
- Marcus Tullius Cicero. 1914. De Finibus Bonorum Et Malorum. Loeb classical library. W. Heinemann and Macmillan.
- Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural Yes/No questions. In Conference of the North American Chapter of the Association for Computational Linguistics.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try ARC, the AI2 reasoning challenge.

- Charles L A Clarke, Gordon V Cormack, and Thomas R Lynam. 2001. Exploiting redundancy in question answering. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '01, pages 358–365, New York, NY, USA. Association for Computing Machinery.
- Cyril Cleverdon. 1967. The cranfield tests on index language devices. In Aslib proceedings. MCB UP Ltd.
- Cyril W Cleverdon. 1991. The significance of the cranfield tests on index languages. In Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '91, pages 3–12, New York, NY, USA. Association for Computing Machinery.
- Adam Coates, Pieter Abbeel, and Andrew Y Ng. 2008. Learning for control from multiple demonstrations. In Proceedings of the International Conference of Machine Learning.
- Kenneth Mark Colby. 1981. Modeling a paranoid mind. The Behavioral and brain sciences, 4(4):515–534.
- Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In Proceedings of the International Conference of Machine Learning.
- Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $\&!#\ast$ vector: Probing sentence embeddings for linguistic properties. In Proceedings of the Association for Computational Linguistics.
- Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popović, and Foldit Players. 2010. Predicting protein structures with a multiplayer online game. Nature, 466(7307):756–760.
- Ann Copestake and Karen Sparck Jones. 1990. Natural language interfaces to databases. Knowledge Engineering Review, 5(4):225–249.
- Charles Corbière, Nicolas Thome, Avner Bar-Hen, Matthieu Cord, and Patrick Pérez. 2019. Addressing failure prediction by learning model confidence. In Proceedings of Advances in Neural Information Processing Systems.
- U.S. Supreme Court. 1927. *Buck v. Bell*. 274.
- Matt Crane. 2018. Questionable answers in question answering research: Reproducibility and variability of published results. Transactions of the Association for Computational Linguistics, 6:241–252.
- Common Crawl. Statistics of common crawl monthly archives.

- Bruce Croft. 2019. Approaches to research in IR. Invited Lecture at the 12th European Summer School in Information Retrieval.
- J Shane Culpepper, Fernando Diaz, and Mark D Smucker. 2018. Research frontiers in information retrieval: Report from the third strategic workshop on information retrieval in lorne (SWIRL 2018). SIGIR Forum, 52(1):34–90.
- Jeffrey Dalton, Chenyan Xiong, Vaibhav Kumar, and Jamie Callan. 2020. CAsT-19: A dataset for conversational information seeking. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1985–1988, New York, NY, USA. Association for Computing Machinery.
- Jeffrey Stephen Dalton, Chen-Yan Xiong, and James P. Callan. 2019. TREC CAsT 2019: The conversational assistance track overview. In Text REtrieval Conference.
- Hoa Trang Dang, Diane Kelly, and Jimmy Lin. 2007. Overview of the trec 2007 question answering track. In Proceedings of the Text REtrieval Conference.
- Hoa Trang Dang, Jimmy Lin, and Diane Kelly. 2006. Overview of the trec 2006 question answering track. In Proceedings of the Text REtrieval Conference.
- Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2018. Multi-step Retriever-Reader interaction for scalable open-domain question answering. In Proceedings of the International Conference on Learning Representations.
- Pradeep Dasigi, Nelson F Liu, Ana Marasović, Noah A Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In Proceedings of Empirical Methods in Natural Language Processing.
- Hal Daumé, Nikos Karampatziakis, John Langford, and Paul Mineiro. 2017. Logarithmic time one-against-some. In Proceedings of the International Conference of Machine Learning.
- Hal Daume III. 2007. Frustratingly easy domain adaptation. In Proceedings of the Association for Computational Linguistics.
- Matt Davis. 2003. Aocdrnig to a rscheearch at cmabrigde uinervtisy. <https://www.mrc-cbu.cam.ac.uk/people/matt.davis/cmabridge/>. Accessed: 2021-03-22.
- Rafael Jaime De Ayala. 2013. The Theory and Practice of Item Response Theory. Guilford Publications.
- Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. Journal of machine learning research: JMLR, 7(1):1–30.

- J Deng, W Dong, R Socher, L Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In Computer Vision and Pattern Recognition.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
- Bhuwan Dhingra, Kathryn Mazaitis, and William W Cohen. 2017. Quasar: Datasets for question answering by search and reading.
- Emily Dinan, Samuel Humeau, Bharath Chintagunta, and Jason Weston. 2019a. Build it break it fix it for dialogue safety: Robustness from adversarial human attack. In Proceedings of Empirical Methods in Natural Language Processing.
- Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019b. Wizard of Wikipedia: Knowledge-powered conversational agents. In Proceedings of the International Conference on Learning Representations.
- Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. 2019. Show your work: Improved reporting of experimental results. Association for Computational Linguistics.
- Ravit Dotan and Smitha Milli. 2020. Value-laden disciplinary shifts in machine learning. In Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency.
- Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker’s guide to testing statistical significance in natural language processing. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In Proceedings of the Association for Computational Linguistics.
- Jesse Dunietz, Gregory Burnham, Akash Bharadwaj, Owen Rambow, Jennifer Chu-Carroll, and David Ferrucci. 2020. To test machine comprehension, start by defining comprehension. In Proceedings of the Association for Computational Linguistics.
- Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. 2017. SearchQA: A new Q&A dataset augmented with context from a search engine.
- Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In Proceedings of the Association for Computational Linguistics.

- F Y Edgeworth. 1888. The statistics of examinations. Journal of the Royal Statistical Society, 51(3):599–635.
- Bradley Efron. 1994. An introduction to the bootstrap. Chapman & Hall, New York.
- Steffen Eger, Yang Gao, Maxime Peyrard, Wei Zhao, and Eduard Hovy, editors. 2020. Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems. Association for Computational Linguistics.
- Ahmed Elgohary, Denis Peskov, and Jordan Boyd-Graber. 2019. Can you unpack that? learning to rewrite Questions-in-Context. In Proceedings of Empirical Methods in Natural Language Processing.
- Ahmed Elgohary, Chen Zhao, and Jordan Boyd-Graber. 2018. A dataset and baselines for sequential open-domain question answering. In Proceedings of Empirical Methods in Natural Language Processing.
- Charles Elkan. 2001. The foundations of cost-sensitive learning. In International Joint Conference on Artificial Intelligence, IJCAI’01, pages 973–978, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Jeffrey L. Elman. 1990. Finding structure in time. Cognitive Science, 14:179–211.
- Mihail Eric and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In Proceedings of the Annual SIGDIAL Meeting on Discourse and Dialogue.
- Kawin Ethayarajh and Dan Jurafsky. 2020. Utility is in the eye of the user: A critique of NLP leaderboards. In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- Allyson Ettinger, Sudha Rao, Hal Daumé III, and Emily M. Bender. 2017. Towards linguistically generalizable NLP systems: A workshop and shared task. In In Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems.
- Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In Proceedings of the Association for Computational Linguistics.
- Shi Feng and Jordan Boyd-Graber. 2019. What can ai do for me: Evaluating machine learning interpretations in cooperative play. In International Conference on Intelligent User Interfaces.
- Shi Feng, Eric Wallace, and Jordan Boyd-Graber. 2019. Misleading failures of partial-input baselines. In Proceedings of the Association for Computational Linguistics.

- Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. 2010. Building watson: An overview of the deepqa project. AI Magazine, 31:59–79.
- Jenny Rose Finkel and Christopher D Manning. 2009. Hierarchical Bayesian domain adaptation. In Proceedings of the Association for Computational Linguistics.
- Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In Proceedings of the 2nd Workshop on Machine Reading for Question Answering. Association for Computational Linguistics.
- Karën Fort, Gilles Adda, and K Bretonnel Cohen. 2011. Amazon mechanical turk: Gold mine or coal mine? Computational Linguistics, 37(2):413–420.
- Apache Software Foundation. Lucene. <https://lucene.apache.org/>.
- Batya Friedman, Peter H. Kahn Jr., and Alan Borning. 2008. Value Sensitive Design and Information Systems, chapter 4. John Wiley & Sons, Ltd.
- Batya Friedman and Helen Nissenbaum. 1996. Bias in computer systems. ACM Transactions on Information and System Security, 14(3):330–347.
- Hannah Fry. 2018. Hello World: Being Human in the Age of Algorithms. W. W. Norton.
- Francis Fukuyama. 1995. Confucianism and democracy. Journal of Democracy, 6(2):20–33.
- Meredith D Gall. 1970. The use of questions in teaching. Review of educational research, 40(5):707–721.
- Howard Gardner. 2011. Frames of Mind: The Theory of Multiple Intelligences. Basic Books.
- Matt Gardner, Yoav Artzi, Victoria Basmova, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hanna Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020a. Evaluating models’ local decision boundaries via contrast sets. In Findings of the Association for Computational Linguistics: EMNLP. Association for Computational Linguistics.

- Matt Gardner, Jonathan Berant, Hannaneh Hajishirzi, Alon Talmor, and Sewon Min. 2020b. Question answering is a format; when is it useful? In Proceedings of the Association for Computational Linguistics.
- Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform.
- Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. 2018. Datasheets for datasets.
- Steven Gelb, Garland E Allen, Andrew Futterman, and Barry A Mehler. 1986. Rewriting mental testing history: The view from the american psychologist. In Sage Race Relations Abstracts, volume 11, pages 18–31.
- Sean M Gerrish and David M Blei. 2011. Predicting legislative roll calls from text. In Proceedings of the International Conference of Machine Learning.
- Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In Proceedings of Empirical Methods in Natural Language Processing.
- Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Scott Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In Association for the Advancement of Artificial Intelligence.
- R Girshick, J Donahue, T Darrell, and J Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In Computer Vision and Pattern Recognition.
- Mark E Glickman and Albyn C Jones. 1999. Rating the chess rating system. Chance, 12.
- Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In Proceedings of the Association for Computational Linguistics.
- Uri Gneezy and Aldo Rustichini. 2000. Pay enough or don’t pay at all. The quarterly journal of economics, 115(3):791–810.
- Henry Herbert Goddard. 1920. Human efficiency and levels of intelligence. Princeton University Press.
- Karan Goel, Nazneen Rajani, Jesse Vig, Samson Tan, Jason Wu, Stephan Zheng, Caiming Xiong, Mohit Bansal, and Christopher Ré. 2021. Robustness gym: Unifying the NLP evaluation landscape.

- Harvey Goldstein. 2012. Francis galton, measurement, psychometrics and social progress. Assessment in Education: Principles, Policy & Practice, 19(2):147–158.
- Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. Deep Learning. MIT Press. <http://www.deeplearningbook.org>.
- Karthik Gopalakrishnan, Behnam Hedayatnia, Qinlang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, and Dilek Hakkani-Tür. 2019. Topical-chat: Towards knowledge-grounded open-domain conversations. In Proceedings of the Annual Conference of the International Speech Communication Association.
- Karthik Gopalakrishnan, Behnam Hedayatnia, Longshaokan Wang, Yang Liu, and Dilek Hakkani-Tur. 2020. Are neural Open-Domain dialog systems robust to speech recognition errors in the dialog history? an empirical study. In Proceedings of the Annual Conference of the International Speech Communication Association.
- Clinton Gormley and Zachary Tong. 2015. Elasticsearch: The Definitive Guide. " O'Reilly Media, Inc."
- Stephen Jay Gould. 1981. The Mismeasure of Man. W. W. Norton & Company.
- Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. International Journal of Computer Vision, 127:398–414.
- Samuel Gratzl, Alexander Lex, Nils Gehlenborg, Hanspeter Pfister, and Marc Streit. 2013. LineUp: visual analysis of multi-attribute rankings. IEEE transactions on visualization and computer graphics, 19(12):2277–2286.
- Bert F Green, Alice K Wolf, Carol Chomsky, and Kenneth Laughery. 1961. Baseball: an automatic question-answerer. In Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference. Association for Computing Machinery.
- Terry Gross. 2016. The supreme court ruling that led to 70,000 forced sterilizations. Fresh Air.
- Anupam Guha, Mohit Iyyer, Danny Bouman, and Jordan Boyd-Graber. 2015. Removing the training wheels: A coreference dataset that entertains humans and challenges computers. In Conference of the North American Chapter of the Association for Computational Linguistics.
- J P Guilford. 1954. Psychometric methods. McGraw-Hill, New York, NY, US.
- Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In Proceedings of the International Conference of Machine Learning.

- Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In Proceedings of Empirical Methods in Natural Language Processing.
- Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In Conference of the North American Chapter of the Association for Computational Linguistics.
- Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-Augmented language model Pre-Training. In Proceedings of the International Conference of Machine Learning.
- A Halevy, P Norvig, and F Pereira. 2009. The unreasonable effectiveness of data. IEEE intelligent systems, 24(2):8–12.
- Ronald Hambleton. 1991. Fundamentals of item response theory. Sage Publications, Newbury Park, Calif.
- Ray Hamel. 2000. Trivia master bio. Accessed: 2021-03-04.
- Sangdo Han, Jeesoo Bang, Seonghan Ryu, and Gary Geunbae Lee. 2015. Exploiting knowledge base to generate responses for natural language dialog listening agents. In Proceedings of the Annual SIGDIAL Meeting on Discourse and Dialogue.
- Elizabeth Hardwick. 1967. The Little Foxes revived. The New York Review of Books, 9(11).
- Stevan Harnad. 1992. The turing test is not a trick: Turing indistinguishability is a scientific criterion. SIGART Bull., 3(4):9–10.
- Bob Harris. 2006. Prisoner of Trebekistan: a decade in Jeopardy!
- Zellig S Harris. 1954. Distributional structure. Word, 10(2-3):146–162.
- He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. 2017. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. In Proceedings of the Association for Computational Linguistics.
- He He, Jordan Boyd-Graber, and Hal Daumé III. 2016a. Interpretese vs. translationese: The uniqueness of human strategies in simultaneous interpretation. In Conference of the North American Chapter of the Association for Computational Linguistics.
- He He, Jordan L. Boyd-Graber, Kevin Kwok, and Hal Daumé III. 2016b. Opponent modeling in deep reinforcement learning. In Proceedings of the International Conference of Machine Learning.

- Behnam Hedayatnia, Karthik Gopalakrishnan, Seokhwan Kim, Yang Liu, Mihail Eric, and Dilek Hakkani-Tur. 2020. Policy-Driven neural response generation for Knowledge-Grounded dialogue systems. In Proceedings of the 13th International Conference on Natural Language Generation.
- Dan Hendrycks and Kevin Gimpel. 2017a. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In Proceedings of the International Conference on Learning Representations.
- Dan Hendrycks and Kevin Gimpel. 2017b. Gaussian error linear units (GELUs).
- Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. 2021. Natural adversarial examples. In Computer Vision and Pattern Recognition.
- Ralf Herbrich, Tom Minka, and Thore Graepel. 2007. Trueskill™: A bayesian skill rating system. In Proceedings of Advances in Neural Information Processing Systems.
- Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In Proceedings of Advances in Neural Information Processing Systems.
- Jose Hernandez-Orallo. 2020. AI evaluation: On broken yardsticks and measurement scales. In Workshop on Evaluating Evaluation of Ai Systems at AAAI.
- William Hersh, Andrew Turpin, Susan Price, Benjamin Chan, Dale Kramer, Lynetta Sacherek, and Daniel Olson. 2000. Do batch and user evaluations give the same results? In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval.
- Daniel Hewlett, Alexandre Lacoste, Llion Jones, Illia Polosukhin, Andrew Fandrianto, Jay Han, Matthew Kelcey, and David Berthelot. 2016. Wikireading: A novel large-scale language understanding task over wikipedia. In Proceedings of the Association for Computational Linguistics.
- Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The goldilocks principle: Reading children’s books with explicit memory representations. In Proceedings of the International Conference on Learning Representations.
- Lynette Hirschman, Marc Light, Eric Breck, and John D Burger. 1999. Deep read: a reading comprehension system. In Proceedings of the Association for Computational Linguistics.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural Computation, 9:1735–1780.

- Kenneth Holstein, Jennifer Wortman Vaughan, Hal Daumé, Miro Dudik, and Hanna Wallach. 2019. Improving fairness in machine learning systems: What do industry practitioners need? In International Conference on Human Factors in Computing Systems, New York, NY, USA. Association for Computing Machinery.
- Mark Hopkins and Jonathan May. 2013. Models of translation competitions. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Yufang Hou, Charles Jochim, Martin Gleize, Francesca Bonin, and Debasis Ganguly. 2019. Identification of tasks, datasets, evaluation metrics, and numeric scores for scientific leaderboards construction. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Dirk Hovy and Shannon L Spruit. 2016. The social impact of natural language processing. In Proceedings of the Association for Computational Linguistics.
- Feng hsiung Hsu, Murray Campbell, and A. Joseph Hoane. 1995. Deep blue system overview. In Proceedings of the 9th International Conference on Supercomputing.
- Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In Proceedings of Empirical Methods in Natural Language Processing.
- Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference of Machine Learning.
- Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. 2014. A neural network for factoid question answering over paragraphs. In Proceedings of Empirical Methods in Natural Language Processing.
- Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In Proceedings of the Association for Computational Linguistics.
- Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In Conference of the North American Chapter of the Association for Computational Linguistics.
- Matthew Frye Jacobson. 1999. Whiteness of a different color: European immigrants and the alchemy of race. Harvard University Press.
- Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. Intelligent Data Analysis, 6:429–449.
- Ken Jennings. 2006. Brainiac: adventures in the curious, competitive, compulsive world of trivia buffs. Villard.

- Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In Proceedings of Empirical Methods in Natural Language Processing.
- Heinrich Jiang, Been Kim, Melody Guan, and Maya Gupta. 2018. To trust or not to trust a classifier. In Proceedings of Advances in Neural Information Processing Systems.
- Xiaoqian Jiang, Melanie Osl, Jihoon Kim, and Lucila Ohno-Machado. 2012. Calibrating predictive model estimates to support personalized medicine. Journal of the American Medical Informatics Association: JAMIA, 19(2):263–274.
- Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. IEEE Transactions on Big Data.
- Karen Spärck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation, 28(1):11–21:133,525.
- Karen Sparck Jones. 2001. Automatic language and information processing: rethinking evaluation. Natural Language Engineering, 7(1):29–46.
- Lyle V. Jones and David Thissen. 2006. 1 a history and overview of psychometrics. In C.R. Rao and S. Sinharay, editors, Psychometrics, volume 26 of Handbook of Statistics, pages 1–27. Elsevier.
- Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. 1999. An introduction to variational methods for graphical models. Machine Learning, 37(2):183–233.
- Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Proceedings of the Association for Computational Linguistics.
- Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In Proceedings of the European Chapter of the Association for Computational Linguistics.
- Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent convolutional neural networks for discourse compositionality. In Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality.
- K Kalgaonkar, C Liu, Y Gong, and K Yao. 2015. Estimating confidence scores on ASR results using recurrent neural networks. In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).
- Amita Kamath, Robin Jia, and Percy Liang. 2020. Selective question answering under domain shift. In Proceedings of the Association for Computational Linguistics.

- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models.
- Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In Proceedings of Empirical Methods in Natural Language Processing.
- Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with Counterfactually-Augmented data. In International Conference on Learning Representations.
- Divyansh Kaushik, Douwe Kiela, Zachary C Lipton, and Wen-Tau Yih. 2021. On the efficacy of adversarial data collection for question answering: Results from a Large-Scale randomized study. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Divyansh Kaushik and Zachary Chase Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In Proceedings of Empirical Methods in Natural Language Processing.
- Greg A Keim, Noam M Shazeer, Michael L Littman, Sushant Agarwal, Catherine M Cheves, Joseph Fitzgerald, Jason Grosland, Fan Jiang, Shannon Pollard, and Karl Weinmeister. 1999. PROVERB: The probabilistic cruciverbalist. In Association for the Advancement of Artificial Intelligence.
- J. F. Kelley. 1984. An iterative design methodology for user-friendly natural language office information applications. ACM Trans. Inf. Syst., 2:26–41.
- M G Kendall. 1938. A new measure of rank correlation. Biometrika, 30(1/2):81–93.
- Ibram X. Kendi. 2016. Stamped from the Beginning: The Definitive History of Racist Ideas in America. PublicAffairs.
- Ibram X Kendi. 2020. Read ibram x. kendi’s testimony in support of the working group recommendation to #suspendthetest. Accessed: 2021-07-16.
- Hamed Khanpour, Nishitha Guntakandla, and Rodney Nielsen. 2016. Dialogue act classification in domain-independent conversations using a deep recurrent neural network. In Proceedings of International Conference on Computational Linguistics.
- Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In Conference of the North American Chapter of the Association for Computational Linguistics.

- Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval.
- Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. QASC: A dataset for question answering via sentence composition. In Association for the Advancement of Artificial Intelligence.
- Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics.
- Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016a. Character-Aware neural language models. In Association for the Advancement of Artificial Intelligence.
- Young-Bum Kim, Karl Stratos, and Ruhi Sarikaya. 2016b. Frustratingly easy neural domain adaptation. In Proceedings of International Conference on Computational Linguistics.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In Proceedings of the International Conference on Learning Representations.
- Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. Transactions of the Association for Computational Linguistics, 6:317–328.
- Yehuda Koren. 2009. The bellkor solution to the netflix grand prize. Netflix prize documentation, 81(2009):1–10.
- A. Korin. 2002. Fenomen “Chto? Gde? Kogda?”. Eksmo.
- Peter Kraft, Hirsh Jain, and Alexander M Rush. 2016. An embedding model for predicting Roll-Call votes. In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- Bernhard Kratzwald and Stefan Feuerriegel. 2018. Adaptive document retrieval for deep question answering. In Proceedings of Empirical Methods in Natural Language Processing.
- Klaus Krippendorff. 2004. Content Analysis: an Introduction to its Methodology. Sage: Thousand Oaks, CA. Chapter 11.

- Thomas Kuhn. 2012. The structure of scientific revolutions. The University of Chicago Press, Chicago.
- Volodymyr Kuleshov and Stefano Ermon. 2017. Estimating uncertainty online against an adversary. In Association for the Advancement of Artificial Intelligence, volume 31.
- Brian Kulis. 2012. Metric Learning: A Survey, volume 5.
- Julian Kupiec. 1993. Murax: A robust linguistic approach for question answering using an on-line encyclopedia. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval.
- Stacey Kuznetsov. 2006. Motivations of contributors to wikipedia. SIGCAS Comput. Soc., 36(2):1–es.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. Transactions of the Association for Computational Linguistics, 7:453–466.
- Cody C T Kwok, Oren Etzioni, and Daniel S Weld. 2001. Scaling question answering to the web. In Proceedings of the World Wide Web Conference. ACM Press.
- Michail G Lagoudakis and Ronald Parr. 2003. Reinforcement learning as classification: Leveraging modern classifiers. In Proceedings of the International Conference of Machine Learning.
- Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In Proceedings of Empirical Methods in Natural Language Processing.
- John P Lalor, Hao Wu, Tsendsuren Munkhdalai, and Hong Yu. 2018. Understanding deep learning performance through an examination of test set difficulty: A psychometric case study. In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- John P Lalor, Hao Wu, and Hong Yu. 2016. Building an evaluation scale using item response theory. In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- John P Lalor, Hao Wu, and Hong Yu. 2019. Learning latent parameters without human response patterns: Item response theory with artificial crowds. In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.

- S K Lam, D Pennock, Dan Cosley, and S Lawrence. 2003. 1 billion pages = 1 million dollars? mining the web to play “who wants to be a millionaire?”. In Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence.
- Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In Proceedings of the Association for Computational Linguistics.
- Nicholas Lemann. 2000. The Big Test: The Secret History of the American Meritocracy. Farrar, Straus and Giroux.
- Hector J Levesque. 2014. On our best behaviour. Artificial intelligence, 212(1):27–35.
- Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The winograd schema challenge. In Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning.
- David D Lewis and William A Gale. 1994. A sequential algorithm for training text classifiers. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval. Springer-Verlag.
- Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2020a. Question and answer Test-Train overlap in Open-Domain question answering datasets.
- Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. PAQ: 65 million Probably-Asked questions and what you can do with them.
- Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-augmented generation for knowledge-intensive NLP tasks. In Proceedings of Advances in Neural Information Processing Systems.
- Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure.
- Ping Li, Christopher J. C. Burges, and Qiang Wu. 2008. Learning to rank using classification and gradient boosting. In Proceedings of Advances in Neural Information Processing Systems.
- Xin Li and Dan Roth. 2002. Learning question classifiers. In Proceedings of International Conference on Computational Linguistics.
- Rongzhong Lian, Min Xie, Fan Wang, Jinhua Peng, and Hua Wu. 2019. Learning to select knowledge for response generation in dialog systems. In International Joint Conference on Artificial Intelligence.

- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Text Summarization Branches Out.
- Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question-answering. Natural Language Engineering, 7:343–360.
- Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. 2019. Reasoning over paragraph effects in situations. In Proceedings of the 2nd Workshop on Machine Reading for Question Answering.
- Xiao Ling, Sameer Singh, and Daniel S Weld. 2015. Design challenges for entity linking. Transactions of the Association for Computational Linguistics, 3:315–328.
- Tal Linzen. 2020. How can we accelerate progress towards human-like linguistic generalization? In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. Transactions of the Association for Computational Linguistics, 4:521–535.
- Zachary C. Lipton and Jacob Steinhardt. 2019. Troubling trends in machine learning scholarship. Queue, 17(1).
- Nelson F Liu, Tony Lee, Robin Jia, and Percy Liang. 2021. Can small and synthetic benchmarks drive modeling innovation? a retrospective study of question answering modeling approaches.
- Shuman Liu, Hongshen Chen, Zhaochun Ren, Yang Feng, Qun Liu, and Dawei Yin. 2018. Knowledge diffusion for neural dialogue generation. In Proceedings of the Association for Computational Linguistics.
- Daniel J Lizotte, Omid Madani, and Russell Greiner. 2003. Budgeted learning of Naive-Bayes classifiers. In Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence.
- F M Lord, M R Novick, and Allan Birnbaum. 1968. Statistical theories of mental test scores.
- Paul Lujan and Seth Teitler. 2003. Writing good quizbowl questions: A quick primer.
- Zhiyi Ma, Kawin Ethayarajh, Tristan Thrush, Somya Jain, Ledell Wu, Robin Jia, Christopher Potts, Adina Williams, and Douwe Kiela. 2021. Dynaboard: An evaluation-as-a-service platform for holistic next-generation benchmarking.
- Subash Maddipoti. Subash maddipoti’s tips on question writing. <https://acf-quizbowl.com/documents/subash-maddipotis-tips-on-question-writing/>. Accessed: 2018-12-04.

- Kenny Malone. 2019. How uncle Jamie broke Jeopardy. Planet Money, (912).
- Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. 2014. Offline policy evaluation across representations with applications to educational games. In Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems.
- F Martínez-Plumed and J Hernández-Orallo. 2020. Dual indicators to analyze AI benchmarks: Difficulty, discrimination, ability, and generality. IEEE Transactions on Computational Intelligence in AI and Games, 12(2):121–131.
- Fernando Martínez-Plumed, Ricardo B C Prudêncio, Adolfo Martínez-Usó, and José Hernández-Orallo. 2016. Making sense of item response theory in machine learning. In Proceedings of the Twenty-second European Conference on Artificial Intelligence.
- Fernando Martínez-Plumed, Ricardo B C Prudêncio, Adolfo Martínez-Usó, and José Hernández-Orallo. 2019. Item response theory in AI: Analysing machine learning classifiers at the instance level. Artificial intelligence, 271:18–42.
- Winter Mason and Duncan J Watts. 2009. Financial incentives and the “performance of crowds”. In Proceedings of the ACM SIGKDD Workshop on Human Computation, HCOMP ’09. Association for Computing Machinery.
- James R. McBride. 1976. Bandwidth, fidelity, and adaptive tests. T. J. McConnell, Jr. (Ed.), CAT/C 2 1975: The second conference on computer-assisted test construction. Atlanta GA: Atlanta Public Schools.
- Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. <http://mallet.cs.umass.edu>.
- Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In Proceedings of Advances in Neural Information Processing Systems.
- Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- H. Mehan. 1979. Learning Lessons: Social Organization in the Classroom. Harvard University Press.
- P Melville, M Saar-Tsechansky, F Provost, and R Mooney. 2005. An expected utility approach to active feature-value acquisition. In Fifth IEEE International Conference on Data Mining.

- Jack Merullo, Luke Yeh, Abram Handler, Alvin Grissom, II, Brendan O’Connor, and Mohit Iyyer. 2019. Investigating sports commentator bias within a large corpus of american football broadcasts. In Proceedings of Empirical Methods in Natural Language Processing.
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In Proceedings of Empirical Methods in Natural Language Processing.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In Proceedings of Advances in Neural Information Processing Systems.
- Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. ParlAI: A dialog research software platform. In Proceedings of Empirical Methods in Natural Language Processing.
- Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016a. Key-value memory networks for directly reading documents. In Proceedings of Empirical Methods in Natural Language Processing.
- Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016b. Key-value memory networks for directly reading documents. In Proceedings of Empirical Methods in Natural Language Processing.
- Sewon Min, Jordan L Boyd-Graber, C Alberti, Danqi Chen, Eunsol Choi, M Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, J Palomaki, Colin Raffel, A Roberts, T Kwiatkowski, Patrick Lewis, Y Wu, Heinrich Kuttler, L Liu, Pasquale Minervini, Pontus Stenetorp, Sebastian Riedel, Sohee Yang, Minjoon Seo, Gautier Izacard, F Petroni, L Hosseini, Nicola De Cao, E Grave, Ikuya Yamada, Sonse Shimaoka, Masatoshi Suzuki, Shumpei Miyawaki, S Sato, Ryo Takahashi, J Suzuki, Martin Fajcik, Martin Docekal, Karel Ondrej, P Smrz, Hao Cheng, Y Shen, X Liu, Pengcheng He, W Chen, Jianfeng Gao, Barlas Oğuz, Xilun Chen, V Karpukhin, Stan Peshterliev, Dmytro Okhonko, M Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Wen-Tau Yih. 2021. NeurIPS 2020 EfficientQA competition: Systems, analyses and lessons learned.
- Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In Proceedings of Empirical Methods in Natural Language Processing.
- Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional questions do not necessitate multi-hop reasoning. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019.

- Model cards for model reporting. In Proceedings of the Conference on Fairness, Accountability, and Transparency. Association for Computing Machinery.
- Bhaskar Mitra and Nick Craswell. 2018. An introduction to neural information retrieval. Foundations and Trends® in Information Retrieval, 13(1):1–126.
- Nikita Moghe, Siddhartha Arora, Suman Banerjee, and Mitesh M. Khapra. 2018. Towards exploiting background knowledge for building conversation systems. In Proceedings of Empirical Methods in Natural Language Processing.
- Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. 2018. Methods for interpreting and understanding deep neural networks. In Digital Signal Processing.
- Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In Proceedings of the Association for Computational Linguistics.
- Kathleen E Moreno, C Douglas Wetzel, James R McBride, and David J Weiss. 1984. Relationship between corresponding armed services vocational aptitude battery (ASVAB) and computerized adaptive testing (CAT) subtests. Applied psychological measurement, 8(2):155–163.
- Pramod Kaushik Mudrakarta, Ankur Taly, Mukund Sundararajan, and Kedar Dhamdhere. 2018. Did the model understand the question? In Proceedings of the Association for Computational Linguistics.
- Tamara Munzner. 2014. Visualization Analysis and Design. CRC Press.
- Brad Myers, Scott E Hudson, and Randy Pausch. 2000. Past, present, and future of user interface software tools. ACM Trans. Comput.-Hum. Interact., 7(1):3–28.
- Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In Proceedings of International Conference on Computational Linguistics.
- Adam Najberg. 2018. Alibaba AI model tops humans in reading comprehension.
- Nikita Nangia and Samuel R Bowman. 2019. Human vs. muppet: A conservative estimate of human performance on the GLUE benchmark. In Proceedings of the Association for Computational Linguistics.
- Prathiba Natesan, Ratna Nandakumar, Tom Minka, and Jonathan D Rubright. 2016. Bayesian prior choice in IRT estimation using MCMC and variational bayes. Frontiers in psychology, 7:1422.
- LLC National Academic Quiz Tournaments. 2020. Naqt | press guide. <https://www.naqt.com/nationals/press-guide.jsp>. Accessed: 2020-03-31.

- Preksha Nema and Mitesh M Khapra. 2018. Towards a better metric for evaluating question generation systems. In Proceedings of Empirical Methods in Natural Language Processing.
- Vincent Ng. 2010. Supervised noun phrase coreference research: The first fifteen years. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Khanh Nguyen and Brendan O’Connor. 2015. Posterior calibration and exploratory analysis for natural language processing models. In Proceedings of Empirical Methods in Natural Language Processing.
- Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated MACHine Reading COMprehension dataset. In Proceedings of the NIPS Workshop on Cognitive Computation: Integrating neural and symbolic approaches.
- Viet-An Nguyen, Jordan Boyd-Graber, Philip Resnik, and Kristina Miler. 2015. Tea party in the house: A hierarchical ideal point topic model and its application to republican legislators in the 112th congress. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Malvina Nissim, Lasha Abzianidze, Kilian Evang, Rob van der Goot, Hessel Haagsma, Barbara Plank, and Martijn Wieling. 2017. Last words: Sharing is caring: The future of shared tasks. Computational Linguistics, 43(4):897–904.
- Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Karen Norrgard. 2008. Human testing, the eugenics movement, and irbs. Nature education, 1(1):170.
- Anne Oeldorf-Hirsch, Brent Hecht, Meredith Ringel Morris, Jaime Teevan, and Darren Gergle. 2014. To search or to ask: the routing of information needs between traditional search engines and social networks. In Conference on Computer Supported Cooperative Work and Social Computing.
- Cathy O’Neil. 2016. Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy. Crown.

- Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. 2018. SemEval-2018 task 11: Machine comprehension using common-sense knowledge. In Proceedings of The 12th International Workshop on Semantic Evaluation. Association for Computational Linguistics.
- Naoki Otani, Toshiaki Nakazawa, Daisuke Kawahara, and Sadao Kurohashi. 2016. IRT-based aggregation model of crowdsourced pairwise comparison for evaluating machine translations. In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D Sculley, Sebastian Nowozin, Joshua V Dillon, Balaji Lakshminarayanan, and Jasper Snoek. 2019. Can you trust your model’s uncertainty? evaluating predictive uncertainty under dataset shift. In Proceedings of Advances in Neural Information Processing Systems.
- Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Kreidieh Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 24:694–707.
- D S Pallett. 2003. A look at NIST’S benchmark ASR tests: past, present, and future. In 2003 IEEE Workshop on Automatic Speech Recognition and Understanding (IEEE Cat. No.03EX721), pages 483–488.
- Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In Proceedings of the Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In Proceedings of the Association for Computational Linguistics.
- Carla Parra Escartín, Wessel Reijers, Teresa Lynn, Joss Moorkens, Andy Way, and Chao-Hong Liu. 2017. Ethical considerations in NLP shared tasks. In Proceedings of the First ACL Workshop on Ethics in Natural Language Processing.
- Prasanna Parthasarathi and Joelle Pineau. 2018. Extending neural generative conversational model using external knowledge sources. In Proceedings of Empirical Methods in Natural Language Processing.
- Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A picture of search. In Proceedings of the 1st international conference on Scalable information systems.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison,

- Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In Proceedings of Advances in Neural Information Processing Systems.
- Kayur Patel, James Fogarty, James A Landay, and Beverly L Harrison. 2008a. Examining difficulties software developers encounter in the adoption of statistical machine learning. In Association for the Advancement of Artificial Intelligence.
- Kayur Patel, James Fogarty, James A. Landay, and Beverly L. Harrison. 2008b. Investigating statistical machine learning as a tool for software development. In International Conference on Human Factors in Computing Systems.
- Amandalynne Paullada, Inioluwa Deborah Raji, Emily M Bender, Emily Denton, and Alex Hanna. 2020. Data and its (dis)contents: A survey of dataset development and use in machine learning research. In Proceedings of the NeurIPS 2020 Workshop: ML Retrospectives, Surveys and Meta-analyses.
- Judea Pearl. 1988. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- Ted Pedersen. 2019. Semeval discussions at NAACL 2019. Accessed: 2021-03-15.
- Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. Journal of Machine Learning Research, 12(85):2825–2830.
- Anselmo Peñas, Eduard Hovy, Pamela Forner, Álvaro Rodrigo, Richard Sutcliffe, and Roser Morante. 2013. QA4MRE 2011-2013: Overview of question answering for machine reading evaluation. In Information Access Evaluation. Multilinguality, Multimodality, and Visualization.
- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In Proceedings of Empirical Methods in Natural Language Processing.
- Denis Peskov, Joe Barrow, Pedro Rodriguez, Graham Neubig, and Jordan Boyd-Graber. 2019. Mitigating noisy inputs for question answering. In Proceedings of the Annual Conference of the International Speech Communication Association.
- Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In Proceedings of the Association for Computational Linguistics.

- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In Conference of the North American Chapter of the Association for Computational Linguistics.
- Michael Petrochuk and Luke S. Zettlemoyer. 2018. Simplequestions nearly solved: A new upperbound and baseline approach. In Proceedings of Empirical Methods in Natural Language Processing.
- Geoff Pleiss, Tianyi Zhang, Ethan R Elenberg, and Kilian Q Weinberger. 2020. Identifying mislabeled data using the area under the margin ranking. In Proceedings of Advances in Neural Information Processing Systems.
- Pavlin G. Poličar, Martin Stražar, and Blaž Zupan. 2019. openTSNE: a modular python library for t-sne dimensionality reduction and embedding.
- Keith T Poole and Howard Rosenthal. 1985. A spatial model for legislative roll call analysis. American journal of political science, 29(2):357–384.
- Keith T Poole and Howard Rosenthal. 2017. Ideology & congress: A political economic history of roll call voting, 2 edition. Routledge, London, England.
- Lianhui Qin, Michel Galley, Chris Brockett, Xiaodong Liu, Xiang Gao, Bill Dolan, Yejin Choi, and Jianfeng Gao. 2019. Conversing by reading: Contentful neural conversation with on-demand machine reading. In Proceedings of the Association for Computational Linguistics.
- Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. 2009. Dataset shift in machine learning. The MIT Press.
- Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In Proceedings of the Conference on Human Information Interaction and Retrieval.
- Md Mustafizur Rahman, Mucahid Kutlu, Tamer Elsayed, and Matthew Lease. 2020. Efficient test collection construction via active learning. In Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval. Association for Computing Machinery.
- Howard Raiffa. 1968. Decision analysis: introductory lectures on choices under uncertainty.
- Anand Rajaraman and Jeffrey David Ullman. 2011. Data Mining, page 1–17. Cambridge University Press.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don’t know: Unanswerable questions for squad. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.

- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- Parikshit Ram and Alexander G Gray. 2012. Maximum inner-product search using cone trees. In Knowledge Discovery and Data Mining.
- Karthik Raman, Paul N. Bennett, and Kevyn Collins-Thompson. 2014. Understanding intrinsic diversity in web search: Improving whole-session relevance. ACM Transactions on Information Systems, 32.
- Georg Rasch. 1960. Studies in Mathematical Psychology: I. Probabilistic Models for Some Intelligence and Attainment Tests. Studies in Mathematical Psychology: I. Probabilistic Models for Some Intelligence and Attainment Tests. Nielsen & Lydiche, Oxford, England.
- Keith Rayner, Sarah J White, Rebecca L Johnson, and Simon P Liversedge. 2006. Raeding wrods with jubmled lettres: there is a cost. Psychological science, 17(3):192–193.
- Joseph Reagle and Lauren Rhue. 2011. Gender bias in Wikipedia and Britannica. International Journal of Communication, 5(0).
- Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. 2019. Do ImageNet classifiers generalize to ImageNet? In Proceedings of the International Conference of Machine Learning. PMLR.
- Mark D Reckase. 2009. Multidimensional item response theory models. In Reckase, editor, Multidimensional Item Response Theory, pages 79–112. Springer New York, New York, NY.
- Ajitha Reddy. 2007. The eugenic origins of iq testing: Implications for post-atkins litigation. DePaul Law Rev., 57:667.
- Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. Transactions of the Association for Computational Linguistics, 7:249–266.
- Pengjie Ren, Zhumin Chen, Zhaochun Ren, Evangelos Kanoulas, Christof Monz, and Maarten de Rijke. 2020. Conversations with search engines.
- Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should I trust you?: Explaining the predictions of any classifier. In Knowledge Discovery and Data Mining.
- Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging nlp models. In Proceedings of the Association for Computational Linguistics.

- Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Mark Richards and Eyal Amir. 2007. Opponent modeling in scrabble. In International Joint Conference on Artificial Intelligence.
- Matthew Richardson, Christopher J C Burges, and Erin Renshaw. 2013. Mctest: A challenge dataset for the open-domain machine comprehension of text. In Proceedings of Empirical Methods in Natural Language Processing.
- Frank Rijmen, Francis Tuerlinckx, Paul De Boeck, and Peter Kuppens. 2003. A non-linear mixed model framework for item response theory. Psychological methods, 8(2):185.
- Peter W van Rijn, Sandip Sinharay, Shelby J Haberman, and Matthew S Johnson. 2016. Assessment of fit of item response theory models used in large-scale educational survey assessments. Large-scale Assessments in Education, 4(1):10.
- Stephen E. Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval.
- Pedro Rodriguez, Joe Barrow, Alexander Hoyle, John P. Lalor, Robin Jia, and Jordan Boyd-Graber. 2021. Evaluation examples are not equally informative: How should that change nlp leaderboards? In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Pedro Rodriguez, Paul Crook, Seungwhan Moon, and Zhiguang Wang. 2020. Information seeking in the spirit of learning: A dataset for conversational curiosity. In Proceedings of Empirical Methods in Natural Language Processing.
- Pedro Rodriguez, Shi Feng, Mohit Iyyer, He He, and Jordan Boyd-Graber. 2019. Quizbowl: The case for incremental question answering.
- Anna Rogers. 2019. How the transformers broke nlp leaderboards.
- Anna Rogers. 2021. Changing the world by changing the data. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2021. QA dataset explosion: A taxonomy of NLP resources for question answering and reading comprehension. arXiv preprint arXiv:2107.12708.
- Anna Rogers, Olga Kovaleva, Matthew Downey, and Anna Rumshisky. 2020a. Getting closer to AI complete question answering: A set of prerequisite real tasks. Association for the Advancement of Artificial Intelligence, 34(05):8722–8731.

- Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020b. A primer in BERTology: What we know about how BERT works. Transactions of the Association for Computational Linguistics, 8:842–866.
- Marc-Antoine Rondeau and T J Hazen. 2018. Systematic error analysis of the Stanford question answering dataset. In Proceedings of the Workshop on Machine Reading for Question Answering. Association for Computational Linguistics.
- Stephane Ross and Drew Bagnell. 2010. Efficient reductions for imitation learning. In Proceedings of Artificial Intelligence and Statistics.
- Stephane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to No-Regret online learning. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics.
- Dan Roth, Heng Ji, Ming-Wei Chang, and Taylor Cassidy. 2014. Wikification and beyond: The challenges of entity and concept grounding. In Proceedings of the Association for Computational Linguistics.
- Abhijit Roy. 2016. (In)visible Publics: Television and Participatory Culture in India, pages 201–221.
- Andrew Ruef, Michael Hicks, James Parker, Dave Levin, Michelle L. Mazurek, and Piotr Mardziel. 2016. Build it, break it, fix it: Contesting secure development. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security.
- David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. Nature, 323(6088):533–536.
- Amrita Saha, Rahul Aralikkatte, Mitesh M Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In Proceedings of the Association for Computational Linguistics.
- Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. WinoGrande: An adversarial winograd schema challenge at scale. In Association for the Advancement of Artificial Intelligence.
- G Salton, A Wong, and C S Yang. 1975. A vector space model for automatic indexing. Communications of the ACM.
- M Sanderson and W B Croft. 2012. The history of information retrieval research. Proceedings of the IEEE, 100(Special Centennial Issue):1444–1451.
- Chinnadhurai Sankar and Sujith Ravi. 2019. Deep reinforcement learning for modeling chit-chat dialog with discrete attributes. Proceedings of the Annual SIGDIAL Meeting on Discourse and Dialogue.

- Yuzuru Sato and Takashi Ikegami. 2004. Undecidability in the imitation game. Minds and Machines, 14(2):133–143.
- Frederik Schadd, Sander Bakkes, and Pieter Spronck. 2007. Opponent modeling in Real-Time strategy games.
- David Schlangen. 2020. Targeting the benchmark: On methodology in current natural language processing research.
- Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. IEEE transactions on signal processing: a publication of the IEEE Signal Processing Society, 45(11):2673–2681.
- Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2020. Green AI. Communications of the ACM, 63(12):54–63.
- Rolf Schwitter, Diego Molla, Rachel Fournier, and Michael Hess. 2000. Answer extraction towards better evaluations of NLP systems. In ANLP-NAACL 2000 Workshop: Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems.
- D Sculley, Jasper Snoek, Alexander B Wiltschko, and A Rahimi. 2018. Winner’s curse? on pace, progress, and empirical rigor. In Proceedings of the International Conference on Learning Representations.
- Joao Sedoc, Daphne Ippolito, Arun Kirubakaran, Jai Thirani, Lyle Ungar, and Chris Callison-Burch. 2019. Chateval: A tool for chatbot evaluation. In Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics.
- João Sedoc and Lyle Ungar. 2020. Item response theory for efficient human evaluation of chatbots. In Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems. Association for Computational Linguistics.
- Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? How controllable attributes affect human judgments. In Conference of the North American Chapter of the Association for Computational Linguistics.
- Priyanka Sen and Amir Saffari. 2020. What do models learn from question answering datasets? In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. Proceedings of the International Conference on Learning Representations.

- Iulian Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2015. Building end-to-end dialogue systems using generative hierarchical neural network models. In Association for the Advancement of Artificial Intelligence.
- Burr Settles. 2009. Active Learning Literature Survey. Technical Report 1648, University of Wisconsin–Madison.
- Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity linking with a knowledge base: Issues, techniques, and solutions. IEEE Transactions on Knowledge and Data Engineering, 27:443–460.
- David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Vedavyas Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy P. Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. 2016. Mastering the game of go with deep neural networks and tree search. Nature, 529:484–489.
- Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In Proceedings of the International Conference on Learning Representations.
- Rion Snow, Brendan O’Connor, Daniel Jurafsky, and Andrew Ng. 2008. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In Proceedings of Empirical Methods in Natural Language Processing.
- Paul Solomon. 1997. Conversation in information-seeking contexts: A test of an analytical framework. Library & information science research, 19(3):217–248.
- Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. Proceedings of the ACM International Conference on Information and Knowledge Management.
- Charles Spearman. 1904. “general intelligence,” objectively determined and measured. The American journal of psychology, 15(2):201–292.
- Joel H Spring. 1972. Psychologists and the war: The meaning of intelligence in the alpha and beta tests. History of education quarterly, 12(1):3–15.
- Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15:1929–1958.
- Robert J Sternberg. 1985. Beyond IQ: A Triarchic Theory of Human Intelligence. Cambridge University Press.

- Marilyn Strathern. 1997. ‘improving ratings’: audit in the british university system. European review, 5(3):305–321.
- Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Roland Stuckardt. 2003. Coreference-based summarization and question answering: a case for high precision anaphor resolution. In International Symposium on Reference Resolution.
- Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. 2018. What makes reading comprehension questions easier? In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- Saku Sugawara, Yusuke Kido, Hikaru Yokono, and Akiko Aizawa. 2017. Evaluation metrics for machine reading comprehension: Prerequisite skills and readability. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In Proceedings of Advances in Neural Information Processing Systems.
- Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. 2017. Re-visiting unreasonable effectiveness of data in deep learning era. In International Conference on Computer Vision.
- Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact Task-Agnostic BERT for Resource-Limited devices. In Proceedings of the Association for Computational Linguistics.
- S. Shyam Sundar. 2007. The MAIN model : A heuristic approach to understanding technology effects on credibility. In Miriam J Metzger and Andrew J Flanagan, editors, Digital media, youth, and credibility, pages 73–100. The MIT Press.
- Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In Proceedings of Advances in Neural Information Processing Systems.
- Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. In Proceedings of the International Conference on Learning Representations.

- Oyvind Tafjord, Peter Clark, Matt Gardner, Wen-Tau Yih, and Ashish Sabharwal. 2019. QUAREL: A dataset and models for answering questions about qualitative relationships. Association for the Advancement of Artificial Intelligence, 33:7063–7071.
- Jean Tague-Sutcliffe. 1992. The pragmatics of information retrieval experimentation, revisited. Information processing & management, 28(4):467–490.
- Alon Talmor and Jonathan Berant. 2018. The web as a Knowledge-Base for answering complex questions. In Proceedings of the Association for Computational Linguistics.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Conference of the North American Chapter of the Association for Computational Linguistics.
- Koji Tanaka, Junya Takayama, and Yuki Arase. 2019. Dialogue-act prediction of future responses based on conversation history. In Proceedings of the Association for Computational Linguistics.
- Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2016. Movieqa: Understanding stories in movies through question-answering. In Computer Vision and Pattern Recognition.
- Mortimer Taube. 1965. A note on the pseudo-mathematics of relevance. American documentation, 16(2):69–72.
- David Taylor, Colin McNulty, and Jo Meek. 2012. Your starter for ten: 50 years of University Challenge.
- Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. 2006. Hierarchical dirichlet processes. Journal of the American Statistical Association, 101(476):1566–1581.
- Gerald Tesauro, David Gondek, Jonathan Lenchner, James Fan, and John M. Prager. 2013. Analysis of watson’s strategies for playing jeopardy! Journal of Artificial Intelligence Research, 47:205–251.
- James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2019. The FEVER2.0 shared task. In Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER). Association for Computational Linguistics.
- L L Thurstone. 1973. Primary mental abilities. In H J Eysenck, editor, The Measurement of Intelligence, pages 131–136. Springer Netherlands, Dordrecht.

- Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. 2015. Chainer: a next-generation open source framework for deep learning. In Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems.
- Kirill Trapeznikov and Venkatesh Saligrama. 2013. Supervised sequential classification under budget constraints. 31:581–589.
- Ross E Traub. 1997. Classical test theory in historical perspective. Educational Measurement, 16:8–13.
- Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordani, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In Proceedings of the 2nd Workshop on Representation Learning for NLP.
- A M Turing. 1937. On computable numbers, with an application to the entscheidungsproblem. Proceedings of the London Mathematical Society. Third Series, s2-42(1):230–265.
- Alan M. Turing. 1950. Computers & thought. chapter Computing Machinery and Intelligence, pages 11–35. MIT Press, Cambridge, MA, USA.
- Julián Urbano. 2016. Test collection reliability: a study of bias and robustness to statistical assumptions via stochastic simulation. Information Retrieval Journal, 19(3):313–350.
- U.S. Department of State Office of the Historian. 2021. The immigration act of 1924 (the johnson-reed act). Accessed: 2021-07-29.
- Clara Vania, Phu Mon Htut, William Huang, Dhara Mungra, Richard Yuanzhe Pang, Jason Phang, Haokun Liu, Kyunghyun Cho, and Samuel R. Bowman. 2021. Comparing test sets with item response theory. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of Advances in Neural Information Processing Systems.
- Jerry Vinokurov. How to write questions. <http://hsquizbowl.org/forums/viewtopic.php?f=30&t=3945>. Accessed: 2018-12-04.
- Jerry Vinokurov, Gautam Kandlikar, Gaurav Kandlikar, Matthew Jackson, Ryan Westbrook, and Rob Carson. 2014. 2014-15 packet submission guidelines. <https://acf-quizbowl.com/archives/archived-guidelines/2014-15-packet-submission-guidelines/>. Accessed: 2019-06-23.

- Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, John Quan, Stephen Gaffney, Stig Petersen, Karen Simonyan, Tom Schaul, Hado van Hasselt, David Silver, Timothy P. Lillicrap, Kevin Calderone, Paul Keet, Anthony Brunasso, David Lawrence, Anders Ekermo, Jacob Repp, and Rodney Tsing. 2017. Starcraft II: A new challenge for reinforcement learning.
- F B Von Der Osten, M Kirley, and T Miller. 2017. The minds of many: Opponent modeling in a stochastic game. In International Joint Conference on Artificial Intelligence.
- Ellen M Voorhees. 1998. Variations in relevance judgments and the measurement of retrieval effectiveness. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval. Association for Computing Machinery.
- Ellen M. Voorhees. 2000a. Overview of the trec-9 question answering track. In Proceedings of the Text REtrieval Conference.
- Ellen M Voorhees. 2000b. The TREC-8 question answering track report.
- Ellen M. Voorhees. 2001. Overview of the trec 2001 question answering track. In Proceedings of the Text REtrieval Conference.
- Ellen M. Voorhees. 2002a. Overview of the trec 2002 question answering track. In Proceedings of the Text REtrieval Conference.
- Ellen M Voorhees. 2002b. The philosophy of information retrieval evaluation. In Evaluation of Cross-Language Information Retrieval Systems.
- Ellen M Voorhees. 2003a. Evaluating the evaluation: A case study using the TREC 2002 question answering track. In Conference of the North American Chapter of the Association for Computational Linguistics.
- Ellen M. Voorhees. 2003b. Overview of the trec 2003 question answering track. In Proceedings of the Text REtrieval Conference.
- Ellen M. Voorhees. 2004. Overview of the trec 2004 question answering track. In Proceedings of the Text REtrieval Conference.
- Ellen M. Voorhees. 2019. The Evolution of Cranfield, pages 45–69. Springer International Publishing, Cham.
- Ellen M. Voorhees and Hoa Trang Dang. 2005. Overview of the trec 2005 question answering track. In Proceedings of the Text REtrieval Conference.
- Ellen M Voorhees, Donna K Harman, et al. 2005. TREC: Experiment and evaluation in information retrieval, volume 63. MIT press Cambridge, MA.

- Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval.
- Harm de Vries, Dzmitry Bahdanau, and Christopher Manning. 2020. Towards ecologically valid research on language user interfaces. arXiv preprint arXiv:2007.14435.
- Eric Wallace, Shi Feng, and Jordan Boyd-Graber. 2018. Interpreting neural networks with nearest neighbors. In EMNLP 2018 Workshop on Analyzing and Interpreting Neural Networks for NLP.
- Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019a. Universal adversarial triggers for attacking and analyzing NLP. In Proceedings of Empirical Methods in Natural Language Processing. Association for Computational Linguistics.
- Eric Wallace, Pedro Rodriguez, Shi Feng, and Jordan Boyd-Graber. 2019b. Trick me if you can: Human-in-the-loop generation of adversarial question answering examples. In Transactions of the Association for Computational Linguistics, pages 387–401.
- Eric Wallace, Jens Tuyls, Junlin Wang, Sanjay Subramanian, Matt Gardner, and Sameer Singh. 2019c. AllenNLP interpret: A framework for explaining predictions of NLP models. In Proceedings of Empirical Methods in Natural Language Processing.
- Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. SuperGLUE: A stickier benchmark for General-Purpose language understanding systems. In Proceedings of Advances in Neural Information Processing Systems.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In Proceedings of the International Conference on Learning Representations.
- Aobo Wang, Cong Duy Vu Hoang, and Min-Yen Kan. 2013. Perspectives on crowdsourcing annotations for natural language processing. Language Resources and Evaluation, 47(1):9–31.
- Kuansan Wang and Jan Pedersen. 2011. Review of MSR-Bing web scale speller challenge. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR ’11, pages 1339–1340, New York, NY, USA. Association for Computing Machinery.

- Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R 3: Reinforced ranker-reader for open-domain question answering. In Association for the Advancement of Artificial Intelligence.
- Zijie J. Wang, Dongjin Choi, Shenyu Xu, and Diyi Yang. 2021. Putting Humans in the Natural Language Processing Loop: A Survey. In Proceedings of the First Workshop on Bridging Human-Computer Interaction and Natural Language Processing. Association for Computational Linguistics.
- Ritchie D. Watson. 1996. Lillian Hellman's "The Little Foxes" and the new south creed: An ironic view of southern history. The Southern Literary Journal, 28(2):59–68.
- Stuart Watt. 1996. Naive psychology and the inverted turing test. Psychology, 7(14):463–518.
- David J Weiss. 1982. Improving measurement quality and efficiency with adaptive testing. Applied psychological measurement, 6(4):473–492.
- David J Weiss and G Gage Kingsbury. 1984. Application of computerized adaptive testing to educational problems. Journal of educational measurement, 21(4):361–375.
- Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making neural QA as simple as possible but not simpler. In Conference on Computational Natural Language Learning.
- Wayne Weiten. 2016. Psychology: Themes and Variations. Cengage Learning.
- Joseph Weizenbaum. 1966. ELIZA—a computer program for the study of natural language communication between man and machine.
- Johannes Welbl, Nelson F Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In Proceedings of the 3rd Workshop on Noisy User-generated Text. Association for Computational Linguistics.
- Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. Transactions of the Association for Computational Linguistics, 6:287–302.
- Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Lina Maria Rojas-Barahona, Pei hao Su, Stefan Ultes, David Vandyke, and Steve J. Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In Proceedings of the European Chapter of the Association for Computational Linguistics.
- Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2016. Towards ai-complete question answering: A set of prerequisite toy tasks. In Proceedings of the International Conference on Learning Representations.

- Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. Proceedings of the International Conference on Learning Representations.
- Frank Wilcoxon. 1945. Individual comparisons by ranking methods. Biometrics Bulletin, 1(6):80–83.
- Terry Winograd. 1971. Procedures as a Representation for Data in a Computer Program for Understanding Natural Language. Ph.D. thesis, Massachusetts Institute of Technology.
- Terry Winograd. 1972. Understanding natural language. Cognitive psychology, 3(1):1–191.
- Terry Winograd. 1977. Five lectures on artificial intelligence. In A. Zampolli, editor, Linguistic structures processing, volume 5 of Fundamental studies in computer science, pages 399–520. North Holland.
- Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. Transactions of the Association for Computational Linguistics, 8:183–198.
- W A Woods. 1972. The Lunar Sciences Natural Language Information System: Final Report. Bolt, Beranek and Newman.
- M Wu, R Davis, B Domingue, C Piech, and Noah D Goodman. 2020. Variational item response theory: Fast, accurate, and expressive. In 13th International Conference on Educational Data Mining.
- Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel Weld. 2019. Errudite: Scalable, reproducible, and testable error analysis. In Proceedings of the Association for Computational Linguistics. Association for Computational Linguistics.
- Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. 2018. Zero-shot learning - a comprehensive evaluation of the good, the bad and the ugly. IEEE transactions on pattern analysis and machine intelligence.
- Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2018a. Wikipedia2vec: An optimized tool for learning embeddings of words and entities from wikipedia. In Proceedings of Empirical Methods in Natural Language Processing.
- Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning distributed representations of texts and entities from knowledge base. Transactions of the Association for Computational Linguistics, 5:397–411.
- Ikuya Yamada, Ryuji Tamaki, Hiroyuki Shindo, and Yoshiyasu Takefuji. 2018b. Studio ousia’s quiz bowl question answering system.

- Roman V. Yampolskiy. 2013. Turing Test as a Defining Feature of AI-Completeness, pages 3–17. Springer Berlin Heidelberg, Berlin, Heidelberg.
- An Yang, Kai Liu, Jing Liu, Yajuan Lyu, and Sujian Li. 2018a. Adaptations of ROUGE and BLEU to better evaluate machine reading comprehension task. In Proceedings of the Workshop on Machine Reading for Question Answering, Melbourne, Australia. Association for Computational Linguistics.
- Yi Yang, Wen tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In Proceedings of Empirical Methods in Natural Language Processing.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018b. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In Proceedings of Empirical Methods in Natural Language Processing.
- Tom Young, Erik Cambria, Iti Chaturvedi, Hao Zhou, Subham Biswas, and Minlie Huang. 2017. Augmenting end-to-end dialogue systems with commonsense knowledge. In Association for the Advancement of Artificial Intelligence.
- Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. QANet: Combining local convolution with global self-attention for reading comprehension. In Proceedings of the International Conference on Learning Representations.
- D Yu, J Li, and L Deng. 2011. Calibration of confidence measures in speech recognition. IEEE transactions on audio, speech, and language processing, 19(8):2461–2473.
- Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In Proceedings of the International Conference of Machine Learning.
- Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In Proceedings of Empirical Methods in Natural Language Processing.
- Jiawei Zhang, Yang Wang, Piero Molino, Lezhi Li, and David S Ebert. 2019. Manifold: A Model-Agnostic framework for interpretation and diagnosis of machine learning models. IEEE transactions on visualization and computer graphics, 25(1):364–373.
- Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018a. Personalizing dialogue agents: I have a dog, do you have pets too? In Proceedings of the Association for Computational Linguistics.

- Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018b. ReCoRD: Bridging the gap between human and machine commonsense reading comprehension.
- Chen Zhao, Chenyan Xiong, Xin Qian, and Jordan Boyd-Graber. 2020a. Complex factoid question answering with a free-text knowledge graph. In Proceedings of the World Wide Web Conference.
- Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul Bennett, and Saurabh Tiwary. 2020b. Transformer-xh: Multi-evidence reasoning with extra hop attention. In Proceedings of the International Conference on Learning Representations.
- Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In Proceedings of the International Conference on Learning Representations.
- Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018a. Commonsense knowledge aware conversation generation with graph attention. In International Joint Conference on Artificial Intelligence.
- Kangyan Zhou, Shrimai Prabhumoye, and Alan W. Black. 2018b. A dataset for document grounded conversations. In Proceedings of Empirical Methods in Natural Language Processing.
- Justin Zobel and Alistair Moffat. 2006. Inverted files for text search engines. ACM Comput. Surv., 38:6.
- Valentina Bayer Zubek and Thomas G. Dietterich. 2002. Pruning improves heuristic search for cost-sensitive learning. In Proceedings of the International Conference of Machine Learning.