ABSTRACT

Title of Thesis: INCREMENTAL PREDICTION OF SENTENCE-FINAL VERBS WITH ATTENTIVE RECURRENT NEURAL NETWORKS

Wenyan Li
Master of Science, 2018

Thesis Directed By: Professor Jordan Boyd-Graber
Department of Computer Science

Sentence-final verb prediction has garnered attention both in computational linguistics and psycholinguistics. It is indispensable for understanding human processing of verb-final languages. More recently, it has been used for computational approaches to simultaneous interpretation, i.e. translation in real-time, from verb-final to verb-medial languages. While previous approaches use classical statistical methods including pattern-matching rules, n-gram language models, or a logistic regression with linguistic features, we introduce an attention-based neural model, Attentive Neural Verb Inference for Incremental Language (ANVIIL), to incrementally predict final verbs on incomplete sentences. Our approach better predicts the final verbs in Japanese and German and provides more interpretable explanations of why those verbs are selected.

INCREMENTAL PREDICTION OF SENTENCE-FINAL VERBS WITH ATTENTIVE RECURRENT NEURAL NETWORKS

by

Wenyan Li

Thesis 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
Master of Science
2018

Advisory Committee:
Professor Betash Babadi, Chair
Professor Jordan Boyd-Graber
Professor Alexander Barg

Acknowledgments

First and foremost I'd like to thank my advisor, Jordan Boyd-Graber for giving me an opportunity to work on the research project of interest to me and helping me with professional guidance and practical advices throughout the process. I would also like to thank my committee members: Alexander Barg and Betash Babadi for generously offering their support, guidance, time and good will. Likewise, I owe my sincere gratitude to Alvin Grissom II for providing generous suggestions and support remotely. It has been a pleasure to work with and learn from all of you.

I owe my deepest thanks to my parents who have always understood me and supported me.

I thank all the people in the feet-thinking group for listening to my ideas and giving great feedbacks. I also thank Uzi Vishkin, as well as all the earlier professors who have helped and inspired me in ways great and small in my academic studies.

I'm grateful to all of my family, friends, classmates, roommates and people who have listened to me, supported me, encouraged me, and because of whom my experience during graduate studies has been one that I will always remember and cherish.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

The prediction of linguistic inputs happens in normal everyday life. People predict the upcoming units in speech before they are observed and thus tend to complete a slow speaker's unfinished sentences. Such phenomenon can be accredited to the probabilistic nature of language comprehension [4]. Final verb prediction is fundamental to human language processing in subject-object-verb (SOV) languages, such as German and Japanese, particularly for simultaneous interpretation (SI).

Instead of waiting until the entire sentence is completed, SI requires prediction and translation of the source text units while the interlocutor is speaking. When human simultaneous interpreters translate from an SOV language to an SVO one incrementally—without waiting for the final verb at the end of a sentence—they must use strategies to reduce the lag, or delay, between the time they hear the source words and the time they translate them [5]. One strategy is final verb prediction: since the verb comes late in the source sentence but early in the target translation, if the verb is predicted in advance, it can be translated before it is heard, allowing for a more "simultaneous" (or **monotonic**) translation [4, 6]. Chernov et al. [7] regard the probability prediction of the verbal and semantic structure of proceeding messages as one of the most important factor that contributes to simultaneity in SI. Such skills can be enhanced with specific training and

increases with linguistic proficiency of the interpreters.

Like humans, for machines to perform SOV-SVO simultaneous machine translation (SMT), improved verb prediction affords more monotonic translations. Matsubara et al. [8] develop Japanese-English spoken language translation with early prediction of English verbs based on pattern-matching with dictionaries, Grissom II et al. [9] predict most frequent fifty sentence-final verbs using n-gram language models and learn when to trust and apply the predictions in translation with reinforcement learning. Grissom II et al. [3] conduct human study on verb prediction in a multi-choice manner and recognize the importance of linguistic features like case markers in verb prediction. They further incorporate the features in a logistic regression model on verb-final sentences and improve the prediction accuracy.

While neural models can identify complex patterns from feature-rich datasets [10], limited research has gone into the also important problem of *long-distance* prediction, particularly for sentence-final verbs, wherein predictions must be made with limited information while expecting reliable accuracy. We introduce a neural model, **A**ttentive **N**eural **V**erb **I**nference for **I**ncremental **L**anguage (ANVIIL), for verb prediction, which outperforms previous models at predicting verbs early and consequently offers potential to significant latency reduction in simultaneous translation from SOV to SVO languages.

In the following chapter, we detail the problem of verb prediction in SOV languages with concrete examples, and formulate the problem as sequential classification. Chapter 3 includes an overview of existing methods on verb prediction and reviews effective language modeling and classification techniques that can be applied in solving the problem. In Chapter 4, we introduce an attention-based recurrent model (ANVIIL) for incremental

verb prediction. We implement ANVIIL and evaluate it on both Japanese and German verb-final sentences in Chapter 5. In Chapter 6, we visualize the incremental prediction process of the final-verbs and provide interpretable explanations of why those verbs are selected. Finally, in Chapter 7, we conclude and discuss future work.

Chapter 2: The Problem of Verb Prediction

German Cazeneuve dankte dort den Männern und sagte, ohne deren kühlen Kopf hätte es vielleicht ein “furchtbares Drama” gegeben.
English Cazeneuve thanked the men there and said that without their cool heads there might have been a “terrible drama”.
Japanese また 大和 国 奈良 県 の 葛城山 に 篭り 密教 の 宿 曜秘法 を 習得 し た と も 言わ.
English It also said that he was acquainted with secretly-lodging accommodation secret law in Katsuragiyama in Nara Prefecture of Yamato.

Figure 2.1: An example of the verb position difference between SOV and SVO languages, where the final verb in German and Japanese is expected much earlier in their English translation.

In simultaneous translation between two languages with great syntactic divergence, such as German and Japanese, waiting for the final verb puts a strain on interpreter’s short-term memory, and often leads to information loss and delay in translation. Indeed, Jörg [4] regards verb anticipation as an essential problem for all German-English interpreters. For example, in Figure 2.1, the final verb, “gegeben”, in German is expected to be translated together with “hätte es” as “there has been” in the middle of the English translation. Similarly, the verb “言わ” in the Japanese sentence which appears at the very end is supposed to be translated as “said” and positioned at the beginning of the translation.

Given an SOV sentence, we want to predict the final verb *as soon as possible* in an incremental setting. This consists of two steps: learning information from the incomplete

sequence and making predictions based on available knowledge. The former step involves language modeling and sequence representation, while the latter includes assigning a specific label, in our case, the verb, to the observed sequence. We follow Grissom II et al. [3] and formulate final verb prediction as sequential classification: a sentence is revealed to the classifier incrementally, and the classifier predicts the label (the verb) at each time step.

2.1 Definition of terms

When formulating the problem of verb prediction, specific terminology in linguistics and computational linguistics are used and might not be familiar to all readers. Hence, we include relevant technical terms in this glossary.

Token When processing a sentence, a token, is a string of contiguous characters which is often used as a useful semantic unit.

Preverb For each sentence, we aim to predict the final verb. During processing, the verb that ends the sentence is extracted as target and the sequence of tokens preceding it is regarded as the preverb.

N-gram An N-gram is a continuous sequence of N words given an input text sentence.

Embeddings In natural language processing (NLP), words or characters in the vocabulary are often mapped into vectors that consist of real numbers. Such vector representations are called embeddings. We represent embedding vectors of a word or character with notation $\mathbf{w}$.

Case markers Case markers are often placed after a word to assign case without changing the original. For example, in Japanese, "ga (が)" is used to indicate subject.

POS-*tagging* Part-of-speech (POS) tags are syntactic categories assigned to words. Main categories including noun, verb, pronoun, preposition, adverb, conjunction, participle, and article. Assigning categories to words is referred as POS-tagging.

Simultaneous translation Simultaneous translation is translating in real-time while the interlocutor is speaking instead of waiting after the entire sentence is completed. Simultaneous translation reduces latency of communication while ensuring translation quality. Generally, input of the source language is segmented into units where translation is performed incrementally.

Chapter 3: Background Research

Verb prediction was mostly studied during the process of interpretation both in humans and machines. Empirical research on German verb prediction was conducted with German-English simultaneous interpreters in [4], in which interpreters (both native German or English speakers) are given sentences with multiple anticipation possibilities. The experiment finds that verb prediction is applied in half of the sentences and prediction accuracy increases with interpreters' experience and language proficiency. Matsubara et al. [8] first introduce early verb prediction into Japanese-English simultaneous machine translation (SMT) by predicting verbs in the target language with dictionaries and pattern-matching. Grissom II et al. [9] and Gu et al. [11] use verb prediction in the source language and learn when to trust the predictions with reinforcement learning, while Oda et al. [12] predict syntactic constituents on translation units and do the same. Grissom II et al. [3] study human's ability in incremental verb prediction with multi-choice questions and develop a computational approach using linear classifiers.

Existing neural attention models for sequential classification are commonly trained on complete input sequences [13, 14, 15]. Classification on incomplete sequences, and, specifically, long-distance sentence-final verb prediction for languages remains a difficult and under-explored problem.

3.1 Classical Statistical Methods for Verb Prediction

In this section we review statistical methods in language modeling and sequence classification that are applied in existing approaches for verb prediction.

3.1.1 N-gram Language Modeling

N-gram modeling estimates the probability of a sequence of words with the chain rule of probability. Given preceding words, the conditional probability of the next word in the sequence is approximated with Markov dependency of order $N-1$ [16, 17], i.e. $P(w_n \,|\, w_1^{n-1}) \approx P(w_n \,|\, w_{n-N+1}^{n-1})$. Using chain rule, the probability of the entire input sequence is

$$P(w_1^n) = \prod_{k=1}^{n} P(w_k \,|\, w_1^{k-1}) \tag{3.1}$$

N-gram probabilities can be computed using maximum likelihood estimation (MLE) with smoothing techniques such as Laplace and Kneser-Ney smoothing, etc [18]. Grissom II et al. [9] apply a Kneser-Ney n-gram model to learn the preverb sequence of each verb.

3.1.2 Linear Classifiers

Naive Bayes Naive Bayes is widely used in classification tasks for its simplicity and effectiveness. Given an input sequence $\mathbf{x}$ and target label y, Naive Bayes models assume independence between features in the given class. Labels are estimated through Bayes rule by learning conditional probabilities of the features $P(\mathbf{x}|y)$ and priors $P(y)$ on a large number of training examples for each class [16, 19]. In [9], verb selection is

defined as Bayesian classification, i.e.

$$v_t \equiv arg\max_{v} \prod_{c \in S_t} p(c|v)p(v) \tag{3.2}$$

At time step t a verb is selected according to the likelihood of the N-gram context of the preverb c regarding to each verb and the priors of the verbs.

Logistic regression Unlike generative models such as Naive Bayes, logistic regression discriminates among classes by directly approximating the the probability of the class label given the input features $P(y|\mathbf{x})$ [16]. Grissom II [20] use a logistic classifier in a one-vs-all manner for predict sentence-final verbs. Given a preverb represented with binary features $\mathbf{x}$, the final verb is estimated as

$$v_t \equiv arg\max_{v} P(v|\mathbf{x}) \tag{3.3}$$

Compared to Naive Bayes, such discriminative model with a rich set of features including unigrams, bigrams and case markers is shown to be more effective at predicting the sentence-final verbs.

3.1.3 Challenge of Learning Long-range Dependencies

The major challenge of using statistical models for verb prediction lies in learning long-range dependencies. Both aforementioned linear models are proved to be useful despite independence assumptions, which however precludes long-range dependencies modeling [21, 22]. We can include adjacent context in features in n-gram based approach with a larger n, but such n-grams which consist of a certain length of continuous words or elements are often scant in the vocabulary and therefore introduce sparsity to the features [23]. Hidden Markov models (HMMs), though being able to learn transitions along

sequence transitions, are computationally impractical, as the state space of the model growing exponentially when considering a large context window [22].

3.2 Recurrent Neural Networks for Sequence Modeling

Konieczny and Döring [24] predict verbs by training a simple recurrent network (SRN) to capture dependency relationships between verbs and preceding arguments. Based on SRNs, recurrent neural networks (RNNs) have emerged as a popular and often effective alternative to statistical n-gram-based models for sequential modeling. While taking input at each time step, RNNs are capable of passing context (i.e. the hidden state) from the previous time step to the next, which enables them to encode an input sequence with arbitrary length into a fixed-length vector $\mathbf{h}_t$.

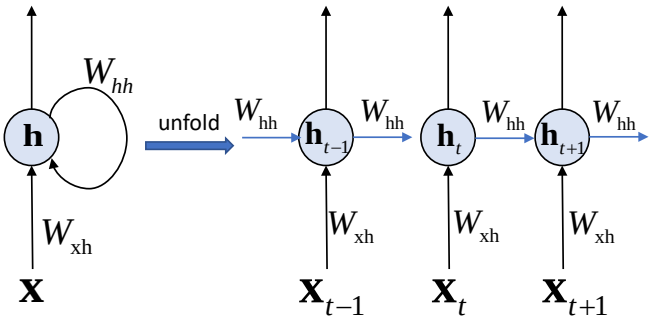


Figure 3.1: A RNN model and its unfolded structure across multiple time steps.

Based on RNNs, a language model can be built as follows[1],

$$\mathbf{h}_t = tanh(W_{xh}\mathbf{x}_t + W_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h) \tag{3.4}$$

[1]here we follow the conventional representation in [25].

where $\mathbf{x}_t$ is the input at time step t, which is often the embedding of the input word or character. The model is parameterized with weights W_{xh}, W_{hh} and bias $\mathbf{b}_h$, which are shared across time steps. All the previous context is expected to be included in the hidden state $\mathbf{h}_{t-1}$, which functions as the memory of the network. The final hidden state $\mathbf{h}_t$ is often regarded as a context representation of the encoded sequence and is further used in classification tasks.

However, during back propagation, the gradients may explode or vanish through time. Gated recurrent networks such as long short-term memory (LSTMs) neural network architecture and gated recurrent units (GRUs) further conquer the problem with their carefully designed architecture [25].

3.2.1 Long Short-Term Memory Network (LSTM)

Unlike vanilla RNNS, LSTM has input, update and forget gates, which allow back propagation through unlimited time steps (Figure 3.2). The architecture of LSTM [26] is

$$\mathbf{u}_t = tanh(W_{xu}\mathbf{x}_t + W_{hu}\mathbf{h}_{t-1} + \mathbf{b}_u) \tag{3.5}$$

$$\mathbf{i}_t = \sigma(W_{xi} + W_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i) \tag{3.6}$$

$$\mathbf{o}_t = \sigma(W_{xo}\mathbf{x}_t + W_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o) \tag{3.7}$$

$$\mathbf{c} = \mathbf{i}_t \odot \mathbf{u}_t + \mathbf{f}_t \odot \mathbf{c}_{t-1} \tag{3.8}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot tanh(\mathbf{c}_t) \tag{3.9}$$

Equation 3.5 calculates the hidden state with the current input in a same way as in regular recurrent network like 3.4, Equation 3.6 and 3.7 are the input and output gates

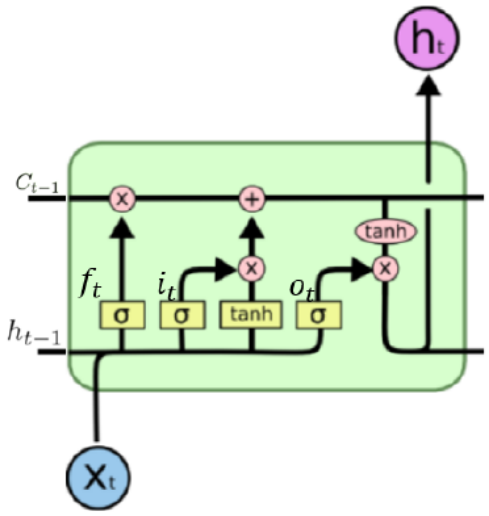


Figure 3.2: Structure of a LSTM model at one single time step (Figure source: [1]).

which control the extent of update and output defined by a sigmoid function. In Equation 3.8, the vanishing/exploding gradient problem is solved by ensuring unit gradient (controlling $\mathbf{f}_t$) and updating the memory cell according to the input gate. Finally, Equation 3.9 gives us the next hidden state of the LSTM.

With LSTMs offer a solution to vanishing gradients problem, we are able to encode the input sequence and use the hidden state to obtain the context representation. However, the architecture of LSTM is rather complex, which can result in computationally ineffectiveness.

3.2.2 Gated Recurrent Units (GRUs)

Recently, gated recurrent units (GRUs) [27] are proved to be an effective alternative to LSTMs for its simplicity in computation and on par performance on various tasks.

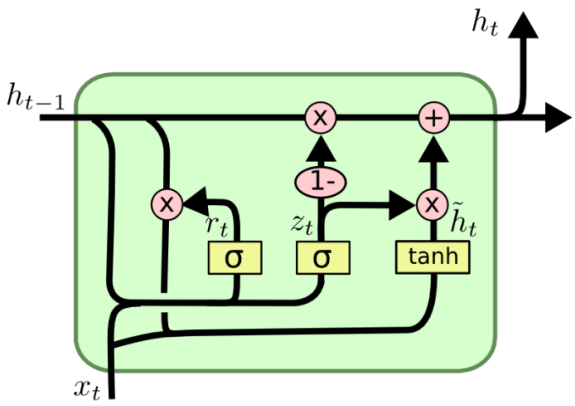


Figure 3.3: Structure of a GRU model at one single time step (Figure source: [2])

The architecture of GRUs in Figure 3.3 is

$$\mathbf{r}_t = \sigma(W_{xr}\mathbf{x}_t + W_{hr}\mathbf{h}_{t-1} + \mathbf{b}_r) \tag{3.10}$$

$$\mathbf{z}_t = \sigma(W_{xz}\mathbf{x}_t + W_{hz}\mathbf{h}_{t-1} + \mathbf{b}_z) \tag{3.11}$$

$$\tilde{\mathbf{h}}_t = tanh(W_{xh}\mathbf{x}_t + W_{hh}(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h) \tag{3.12}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t)\mathbf{h}_{t-1} + \mathbf{z}_t\tilde{\mathbf{h}}_t \tag{3.13}$$

Equation 3.10 and 3.11 are similar to the input and output gates in LSTM in 3.6 and Equation 3.7, where the sigmoid function outputs the extent of reset and update. In Equation 3.13, GRU outputs a hidden state at time step t with respect to the value of the update gate. We rewrite the above equations as $\mathbf{h}_t = \text{GRU}(\mathbf{x}_t, \mathbf{h}_{t-1})$ for further use.

3.2.3 Bidirectional RNNs

Bidirectional RNN is proposed by Schuster and Paliwal [28] for incorporating more available information during sequence encoding. In addition to the forward layer in regular RNNs, a bidirectional RNN has another layer which reads the input sequence reversely, allowing future information to be included at the current time step (Equation 3.15).

$$\overrightarrow{\mathbf{h}_t} = tanh(\overrightarrow{W}_{xh}\mathbf{x}_t + \overrightarrow{W}_{hh}\overrightarrow{\mathbf{h}}_{t-1} + \overrightarrow{\mathbf{b}}_h) \tag{3.14}$$

$$\overleftarrow{\mathbf{h}_t} = tanh(\overleftarrow{W}_{xh}\mathbf{x}_t + \overleftarrow{W}_{hh}\overleftarrow{\mathbf{h}}_{t-1} + \overleftarrow{\mathbf{b}}_h) \tag{3.15}$$

By concatenating the hidden states, past and future context information can be summarized at each time step as $\mathbf{h}_t = [\overrightarrow{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t]$. Accordingly, we can construct bidirectional LSTMs and GRUs by adding an additional backward layer to their original structure.

3.3 Attention-based Sentence Embedding

While RNNs, especially LSTMs and GRUs, are effective at encoding an input with arbitrary length sequentially, they are forced to encode all the elements in the sequence yet some of them may not be relevant or helpful to the task. Moreover, relationships between long-distance dependencies can be hard to capture in the sequential setting. To solve this problem, attention mechanism was first used in RNNs for mimicking human visual focus on image classification [29] and was then successfully applied in neural machine translation (NMT) systems for alignment of the source and target sentences by Bahdanau et al. [15] and Luong et al. [30]. Until recently, self-attention, which focuses on intra-relationships between elements of a single input sequence is proved to be effective in

sentence representation [31], classification [13] and translation tasks [32].

For sequence classification, a common approach is to predict with a context vector, which is often the final hidden state of the RNN or the max-pooling of hidden states over all time steps, of the input sequence. Attention mechanism, however, provides a more efficient representation of the sequence by learning attention weights assigned to the hidden states, which tells the encoder how much it needs to focus on the corresponding word or element. More specifically, if we represent a sequence of length l with its hidden states as $H = (\mathbf{h_1}, \mathbf{h_2}, \cdots, \mathbf{h_l})$, attention mechanism allows us to acquire a context vector $\mathbf{v}$ with attention weights $\mathbf{a} = [a_1, a_2, \ldots, a_l]$ such that

$$\mathbf{v} = \sum_{i=1}^{l} a_i \mathbf{h}_i \tag{3.16}$$

.

3.3.1 Calculating Attention Weights

Methods for calculating attention weights of a single sequence are evolved from those dealing with source and target sentence pairs in neural machine translation (NMT).

Served as a basis for most of the attention computing functions, Luong et al. [30] propose three functions for calculating the attention weights with pairwise sequences:

$$attn_i(\mathbf{h}_{target}, \mathbf{h}_i) = \begin{cases} \mathbf{h}_{target}^T \mathbf{h}_i & \textit{dot} \\ \mathbf{h}_{target}^T W_a \mathbf{h}_{source} & \textit{bilinear} \\ \boldsymbol{\mu}_a^T tanh(W_a[\mathbf{h}_{target}^T; \mathbf{h}_i]) & \textit{multi-layer perceptron} \end{cases} \tag{3.17}$$

where $\mathbf{h}_{target}$ is the hidden state at time t in the target sequence and $\mathbf{h}_i$ is the i-th hidden state in the source sequence. The attention weights $\mathbf{a} = [a_1, a_2, \ldots, a_l]$ is obtained by

computing the attention of target hidden state regarding to each of the hidden states $\mathbf{h}_i \in H_{source}$ in the source and are normalized to values between zero to one with a softmax function among all source hidden states.

Unlike MT systems which predict a sequence of target words and contain multiple target hidden states, only one single label needs to be predicted in sequence classification tasks. Thus, functions for attention weights calculation are modified from Equation 3.17. Shen and Lee [14] compute attention weights assigned to each word in the input by using a *dot product* function to measure the similarity of each word embedding and the last hidden state of LSTM:

$$attn_i = \mathbf{h}_T \odot \mathbf{w}_i \tag{3.18}$$

where $\mathbf{h}_T$ is the last hidden state of LSTM and $\mathbf{w}_i$ is the embedding vector of the i-th word in the input sequence. They then obtain the final attention weight through normalization with a logistic sigmoid function:

$$a_i = \frac{\sigma(attn_i)}{\sum_{i=1}^{T} \sigma(attn_i)} \tag{3.19}$$

Yang et al. [13] applies hierarchical attention at word and sentence level to complete document classification. At both levels, they use a multi-layer perceptron with bias to obtain the attention weights (Equation 3.20) and normalize with a softmax function (Equation 3.21). The first layer in the MLP is for obtaining hidden annotations of the hidden states in BiGRU, while the second computes similarity by considering $\boldsymbol{u}_a$ as a context

vector.

$$attn_i = \boldsymbol{u}_a^T tanh(W_a \mathbf{h}_i + b) \tag{3.20}$$

$$a_i = \frac{exp(attn_i)}{\sum_t exp(attn_i)} \tag{3.21}$$

Lin et al. [31] applies multiple attention views on input sentence by the extending the vector $\mathbf{u}_a$ in the second layer of the MLP into a weight matrix $\mathbf{W}_{a_2}$ of dimension r-by-l. Therefore, instead of obtaining a single attention vector, an attention matrix is acquired (Equation 3.22), which allows r views of the input sequences and encourages hops of attention.

$$A = softmax(\mathbf{W}_{a_2}\text{tanh}(\mathbf{W}_{a_1} H^T)) \tag{3.22}$$

By applying the attention matrix on the hidden states, the final sentence embedding in [31] is:

$$M = AH \tag{3.23}$$

Chapter 4: Methods

Having discussed in Chapter 2 and Chapter 3, final verb prediction can be formulated as sequential classification, where various techniques can be applied in dependency learning and sequence encoding. In this Chapter, we introduce an attention neural model, **A**ttentive **N**eural **V**erb **I**nference for **I**ncremental **L**anguage (ANVIIL), which takes embeddings[1] as inputs and captures relations between tokens, regardless of the distance. Predictions of the final verbs are then made by classifying on the encoded sequences.

4.1 Baseline Model

We use the logistic regression model described in Chapter 3.1.2 as a baseline method. The model uses context features and verb features including token unigrams and bigrams, case marker bigrams, and the last observed case marker. The features are obtained by using a morphological analyzer on the input sentences. A one-vs-all strategy is employed for classifying among multiple verbs, i.e. a binary classifier is trained for each (verb) class with sequences belong to that class labeled as positive ($+1$) and all other samples negative (-1).

[1] Embeddings are learned from scratch, as pretrained embeddings [33] did not improve prediction.

4.2 ANVIIL: A Self-attentive Neural Model for Verb Prediction

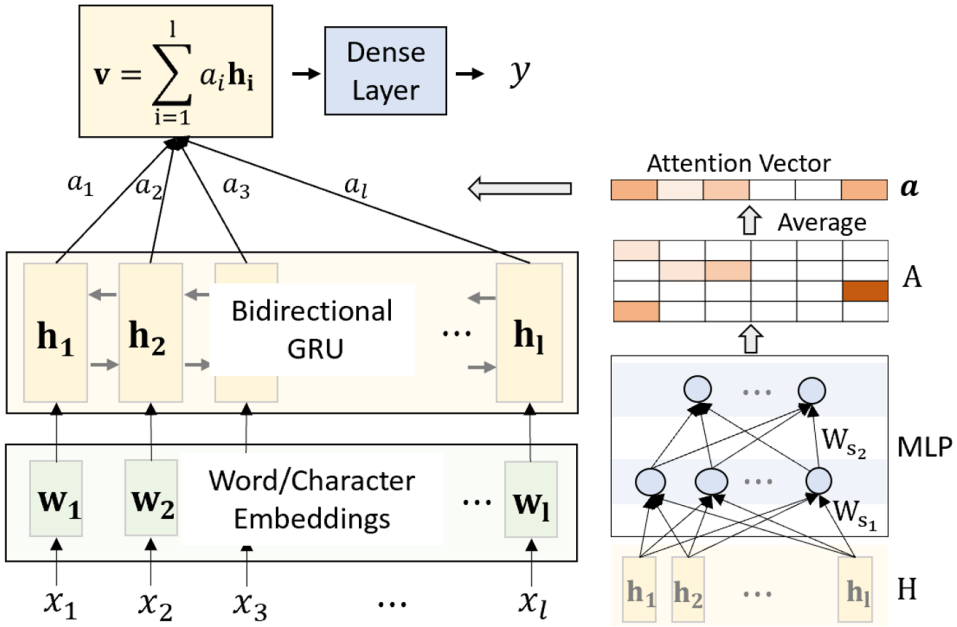


Figure 4.1: ANVIIL. Token sequences at the input layer are mapped to embeddings, which go to the BiGRU. The dot product of attention weights and hidden states pass through a dense layer to predict the verb.

This section describes ANVIIL's structure. As reviewed in Section 3.2, gated recurrent neural networks (RNNs), such as LSTMs and GRUs can effectively encode input sentences of arbitrary length, while self-attention can assist with long-range dependency learning, which we need for effective verb prediction.

We construct an RNN-based classifier with self-attention for predicting sentence-final verbs (Figure 4.1). This is a natural encoding of the problem, as it explicitly models how an interpreter might receive information and update their verb predictions. The hidden states of the sequence model can be either at the word or character level.

4.2.1 BiGRU Sequence Encoder

Recognizing the advantages of GRUs and bi-directional networks as discussed in Chapter 3.2.2 and Chapter 3.2.3, we encode input sequences using bidirectional GRU (BiGRU), which allows later words to change the internal representation of earlier words (impossible with unidirectional RNN). Given an incomplete sentence $\mathbf{x} = (x_1, x_2, \cdots, x_l)$ of length l, BiGRU takes input embeddings $(\mathbf{w_1}, \mathbf{w_2}, \cdots, \mathbf{w_l})$, where $\mathbf{w_i}$ is the d-dimension embedding vector of x_i. At time step t, the forward and backward hidden states are:

$$\begin{aligned} \overrightarrow{h_t} &= \overrightarrow{\text{GRU}}(w_t, \overrightarrow{h}_{t-1}) \\ \overleftarrow{h_t} &= \overleftarrow{\text{GRU}}(w_t, \overleftarrow{h}_{t+1}). \end{aligned} \tag{4.1}$$

These are concatenated as $h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}]$ and we represent the input sequence as:

$$H = (\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_l). \tag{4.2}$$

Embedding vectors for the input can be word embeddings or character embeddings, yielding a word-based or a charactered-based model; we try both and detail in Chapter 5.

4.2.2 Structured Self-attention on Incomplete Sentences

Though self-attention has proved to be successful on sentence embedding and classification, assigning attention weights to incomplete inputs is non-trivial. As described in 3.3, most self-attention mechanisms are applied on tasks including sentence and document classifications given complete input sentences. When applying to incomplete sentences such as segments of the preverbs in our case, attention weights based on similarity measuring or single-layer MLP tends to assign most weights to the latest update token in

the input sequence. It is understandable that the latest word is of the shortest distance to the target and thus has a high probability to be relevant, however, such implementation does not improve over the traditional sequential encoding and we hope to capture relationships between the tokens, regardless of the distance.

Therefore, we apply self-attention with multiple views to allow attention hops on the input sequence. Following Equation 3.22, by using a 2-layer multilayer perceptron (MLP) without bias and a softmax function over the sequence length, we have an r-by-l attention matrix A, which includes r attention vectors extracted from r views of $\mathbf{x}$:

$$A = softmax(\mathbf{W}_{s_2}\text{tanh}(\mathbf{W}_{s_1}H^T)) \tag{4.3}$$

We sum over all r attention vectors and normalize, yielding a single average attention vector $\mathbf{a}$ with normalized weights which summarizes attention from multiple views (Figure 4.1). By assigning each hidden state its attention a_i, we acquire an overall representation of the sequence:

$$\mathbf{v} = \sum_{i=1}^{l} a_i \mathbf{h}_i. \tag{4.4}$$

Unlike the sentence embedding in Equation 3.23, our model applies the attention scores in the averaged vector $\mathbf{a}$. This results in classification with a single context vector $\mathbf{v}$ rather than using a matrix representation, which is shown in Chapter 5 to perform better on the task of verb prediction.

Chapter 5: Experiments and Results

In this Chapter, we evaluate ANVIIL (Section 4.2) against the logistic regression baseline model (Section 4.1) by predicting verbs on both Japanese and German verb-final sentences. Having recognized the sparsity of Japanese verb inflections, we extend the prediction to Japanese verb tenses in addition to predicting the normalized verbs.

5.1 Datasets and Preprocessing

For German, we use the Wortschatz Leipzig news corpus from 2011 to 2015 [34]; for Japanese, we use the Kyoto Free Translation Task (KFTT) corpus of Wikipedia articles. German text is tokenized and POS-tagged with TreeTagger [35]. Digits in the text are replaced with zeros to reduce vocabulary size [36]. We extract sentences ending with a verb tag as samples. Since Japanese is an unsegmented language, we use the morphological analyzer MeCab [37] to tokenize the text. Similarly to Grissom II et al. [3], for Japanese, we strip out post-verbal copulas and normalize the verb forms to the dictionary *ru* (non-past tense) form instead of the fully inflected verb, as Japanese verb inflections are complex, making fully inflected verbs often extremely sparse even in large corpora (Figure 5.3). We also consider *suru* light verb constructions as a single unit.

We obtain 443,054 German and 57,648 Japanese verb-final sentences between ten

Original German Sentence Die Familie war am 12. Juni im Jemen mit zwei deutschen Bibelschülerinnen, einem Briten und einer Südkoreanerin gekidnappt worden.

English The family had been kidnapped in Yemen on June 12 with two German Bible students, one British and one South Korean.

POS-tagged die/ART familie/NN war/VAFIN am/APPRART 12/CARD juni/NN mit/APPR zwei/CARD deutschen/ADJA bibelschülerinnen/NN einem/ART briten/NN und/KON einer/ART südkoreanerin/NN gekidnappt/VVPP worden/VAPP.

Processed die familie war am 00 juni mit zwei deutschen bibelschülerinnen einem briten und einer südkoreanerin gekidnappt (kidnappen).

Figure 5.1: An example of text preprocessing. Original sentence is first POS-tagged, then the sentence-final word with a verb tag is extracted as target. For sentence ends with two continuous verbs, like "gekidnappt worden" in red, the context verb "gekidnappt" is used as target. Digits (highlighted in green) are replaced with zeros. The processed sentence has two components: the preverb (in blue) and the target verb, where the target can be either inflected verb (highlighted in red) or legitimatized verb (in orange).

and fifty tokens long ending in the 100 most common verbs [38], with average sentence lengths of twenty-four and eighteen, respectively. For each sentence, we extract the verb that ends the sentence as the label; the sequence of tokens preceding it (the **preverb**) is the input. Since German sentences may end with two verbs—for example, a verb followed by *ist*, we only predict the content verb in these cases, i.e., the first verb in the two-verb sequence (Figure 5.1). We split the sentences into train (64%), evaluation (16%) and test (20%) sets, yielding 88,611 German and 11,530 Japanese test sentences.

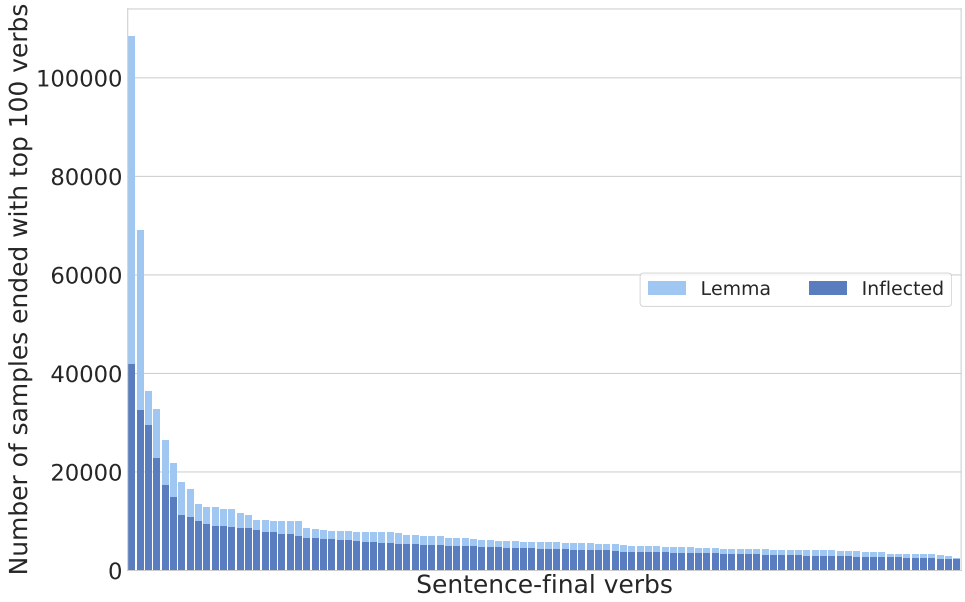


Figure 5.2: Distribution of most frequent 100 German verbs (inflected and normalized)

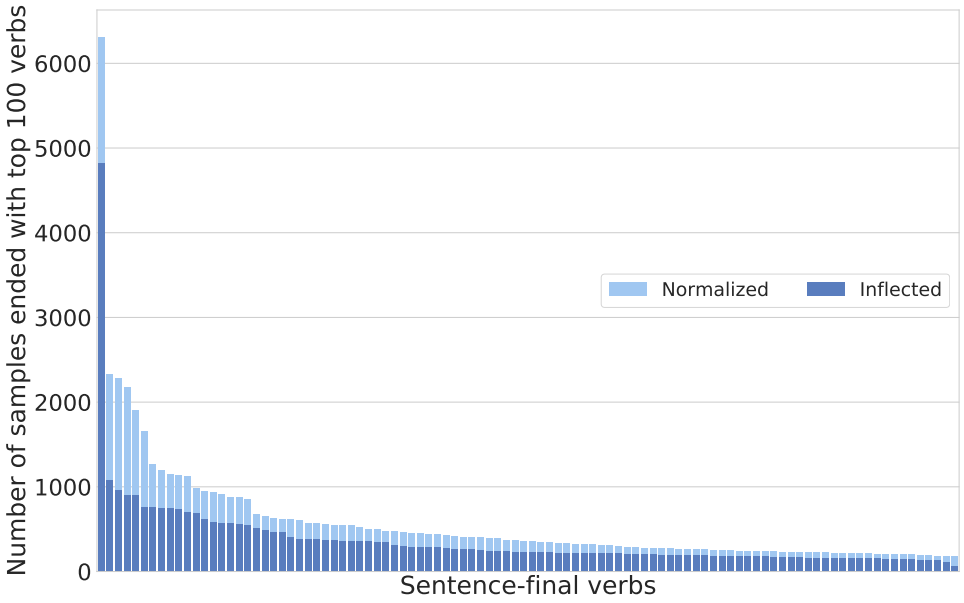


Figure 5.3: Distribution of most frequent 100 Japanese verbs (inflected and normalized). Note that the number of samples using inflected verbs as target is almost halved for most frequent 50 verbs compared to normalized verbs.

5.2 Training Data Representation

We must predict based on incomplete inputs; so, we train on subsequences of each preverb. Each subsequence is an independent input sample during training, and each preverb is truncated into five progressively longer subsentences with length of 30%, 50%, 70%, 90%, and 100% of the original.

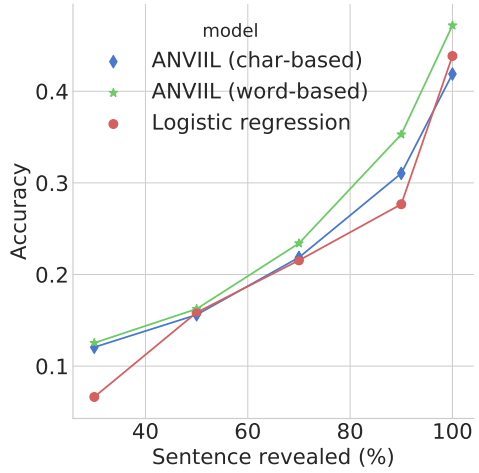
As input sequences vary in length, for computational efficiency, we pad input samples with 0's and train in minibatches similarly to neural MT tasks [39, 40].

5.3 Training Details

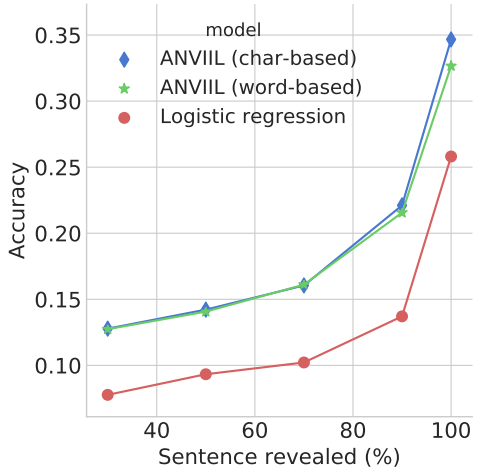
We train both word-based and character-based models for German and Japanese verb prediction.

Character-based Model For input character sequences, we learn 64-dimensional embeddings and encode them with a 2-layer BiGRU of 256 hidden units. Mini-batch sizes are 256 for German and reduced to 128 for Japanese due to the smaller corpus size. We use the evaluation set for tuning and set the embedding dropout rate as 0.6 and the RNN dropout rate as 0.2 while averaging from five views for attention vectors. We optimize with Adam [41] with an initial learning rate of 10^{-4}, decaying by 0.1 when loss increases.

Word-based Model We use a vocabulary size of 50K for German and Japanese input; we use the $<$*UNK*$>$ token for out-of-vocabulary tokens. The embedding size is 300. For German, we encode the input embeddings with a 2-layer BiGRU with 512 hidden units. We reduce the hidden layer size to 256 for Japanese. Other hyperparameters are



(a) German: predicting inflected verbs



(b) Japanese: predicting normalized verbs

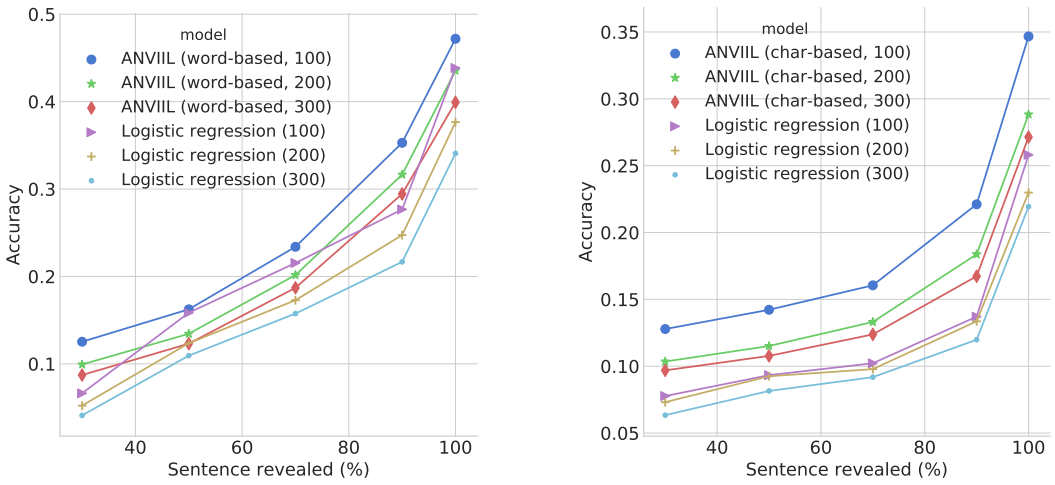
Figure 5.4: Verb prediction accuracy across sentence positions on the 100 most common verbs for German and Japanese. ANVIIL consistently outperforms the baseline model described in Grissom II et al. [3].

unchanged from the character-based model.

5.4 Results

We compare ANVIIL to the logistic regression model described in Section 4.1 on the 100 most common verbs (Figure 5.4).

For both languages, ANVIIL outperforms previous work, more accurately selecting from hundreds of verbs (Figure 5.5), especially early in the sentences. With more words revealed, accuracy of verb prediction increases. The character-based model works best for Japanese. Results of ANVIIL compared to BiLSTM and the baseline logistic regression model are summarized in Table 5.1.



(a) German: predicting inflected verbs

(b) Japanese: predicting normalized verbs

Figure 5.5: Accuracy when classifying among the most common 100, 200, and 300 verbs. ANVIIL consistently outperforms the best-performing model described in Grissom II et al. [3], especially early in the sentences.

5.5 Tense Prediction on Japanese

Due to the sparsity of Japanese inflected verbs, we predict normalized Japanese verbs in Section 5.4, which offers no tense-related information. Hence, we also experiment using ANVIIL to predict the tense of the final verb from the input subsentence. The problem is further formulated as binary classification, i.e. predicting whether the unseen verb is in present or past tense.

5.5.1 Label Extraction and Training Details

We extract tense information from sentences ending in most frequent 500 verbs of the KFTT Japanese dataset, and obtain input sequences in a similar manner as in verb pre-

Model	Japanese		German	
	Avg	End	Avg	End
Logistic regression (Grissom II et al. [9])	0.166	0.382	0.212	0.476
BiLSTM + attn (Lin et al. [31])	0.235	0.400	0.316	0.534
BiGRU	0.229	0.390	0.309	0.521
ANVIIL	0.242	0.401	0.320	0.548

Table 5.1: Verb prediction accuracy (lemmatized) of different models on Japanese and German datasets averaging over all time steps and at the end position. Use character-based model for Japanese and word-based model for German.

diction (Section 5.3), while taking tense classes including"present" and "past", as targets. In Japanese, "た" is often appended to the end of the verb for indicating the past tense. Based on this rule, we extract the tense class based on the "た" indicator of the inflected verbs. With $102,856$ complete Japanese sentences in total, we obtain $329,135$ training, $82,285$ evaluation, and $102,860$ testing samples, respectively.

ANVIIL makes more accurate verb prediction on Japanese (Section 5.4) when modeling the input sequence on a character-based level. Therefore, despite the difference in target labels, the model is trained in the same way as training character-based model for verb prediction. Hyperparameters also remain identical as in the character-based model in 5.3.

5.5.2 Results on Incremental Tense Prediction

As in Figure 5.6, we show that ANVIIL is also capable of predicting the tense of the target verb from revealed subsentence. Like verb prediction, the prediction accuracy grows when more information of the input subsentence is revealed to the model and achieves 70.39% at the end. The success on predicting tense of the sentence-final verbs proves that ANVIIL is able to learn syntactic dependencies in the process of sequence modeling. We also visualize the prediction process by plotting attention heat maps in Figure 6.3.

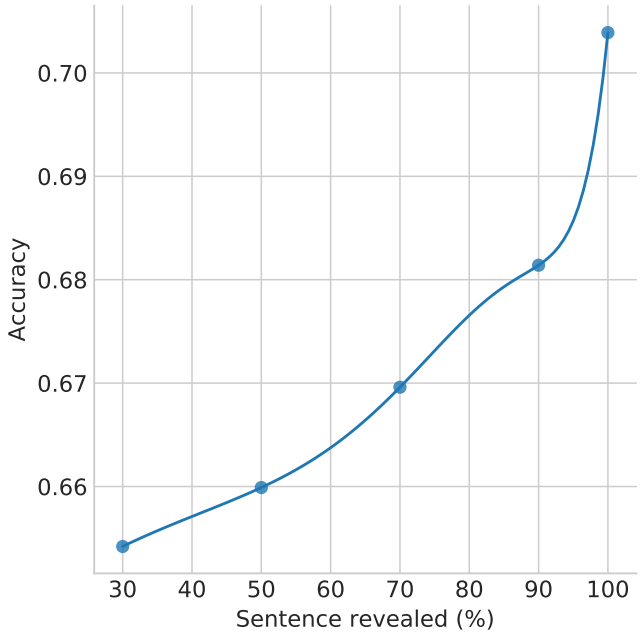


Figure 5.6: Accuracy of incremental tense prediction on Japanese

Chapter 6: Visualization and Discussion

6.1 Visualization

For verb prediction, which we formulate as a classification problem on incomplete sentences, self-attention helps with long-distance dependency learning and context representation on top of the RNNs. Meanwhile, by keeping records of the attention weights assigned to tokens in each sequence, self-attention offers interpretable explanation of the prediction process and is easy to visualize.

We visualize how our model makes its predictions in Figure 6.1 and Figure 6.2. In both languages, the model not only focuses on the most recent revealed word, but also pays attention to dependencies in distance that are considered relevant. While the latest updated word at each time step is of high probability being related to the target verb on a local level, attention on long-term relevance indicates that the model is capable of capturing linguistic cues to make the correct prediction.

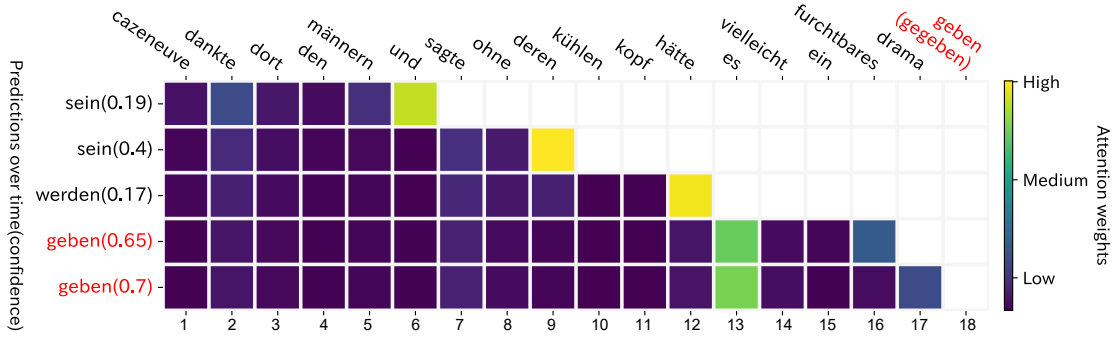


Figure 6.1: Attention and prediction transition through time on a German sentence. The model usually attends to the most recent word, but is able to continually focus on "es", which can be used as the *subject* of an existential phrase [42] in combination with the verb "geben".

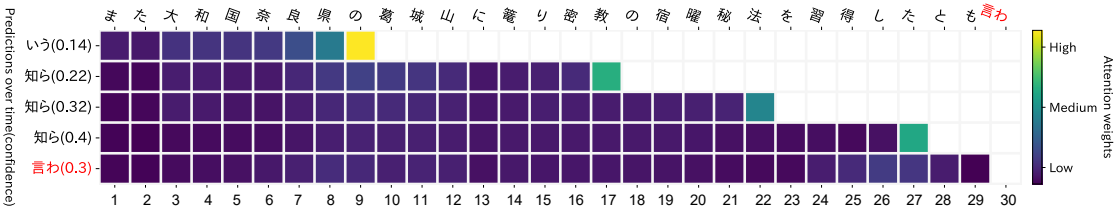


Figure 6.2: Attention and prediction transition through time on a Japanese sentence. The genitive case marker *no*, in bright yellow, has a high attention weight, as do the characters making up the noun preceding it. There is significant attention on case marker-adjacent nouns, including before the genitive *no* (twice) and the accusative *wo*. Toward the end of the sentence, we also see significant attention weights on the quotative particle *to*, which significantly limits possible completions.

6.2 Discussion

6.2.1 Character-based versus Word-based

As described in Section 5.3, we implement both character-based and word-based model for verb prediction. For Japanese final-verb prediction, character-based model has

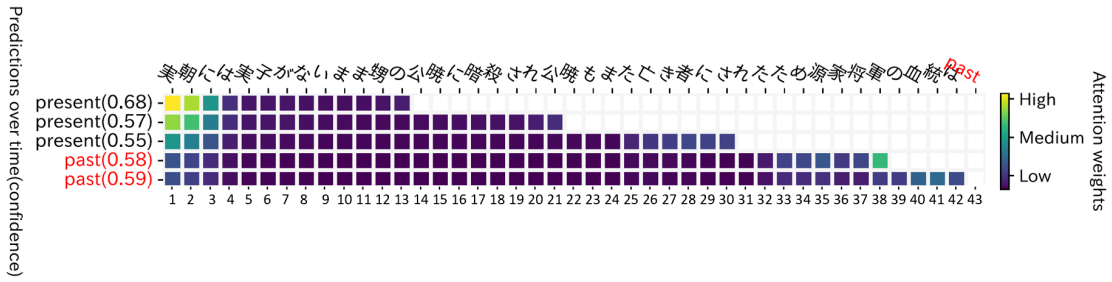


Figure 6.3: Transition of attention weights and verb tense prediction through time on a Japanese sentence. The model first predicts present tense by focusing on "実 朝", which means "in the morning" before half of the sentence is revealed. With more information available, it transfers attention to the latter part of the sentence and correctly predicts past tense.

higher prediction accuracy. Moreover, compared to word-based model, it does not require morphological analyzer and has a smaller vocabulary size. The word-based model, however, works better for German verb prediction and the attention heatmaps of which are more interpretable. We show the contrast in Figure 6.4 and Figure 6.5. Such phenomenon may be explained by difference between Japanese and German writing systems as the former has a very rich agglutinative conjugation system while the latter does not.

6.2.2 Lemmatized verbs versus Inflected verbs

Tense and modality are appended in a format of grammatical markers after verbs in Japanese. Grissom II et al. [3] include case markers as linguistic features during classification. While verb stemming plays an important role of reducing sparsity in Japanese verb prediction, difference in accuracy of predicting inflected and lemmatized final-verbs in German are negligible (Figure 6.6).

Though information regarding to tense of the Japanese verbs are lost by only pre-

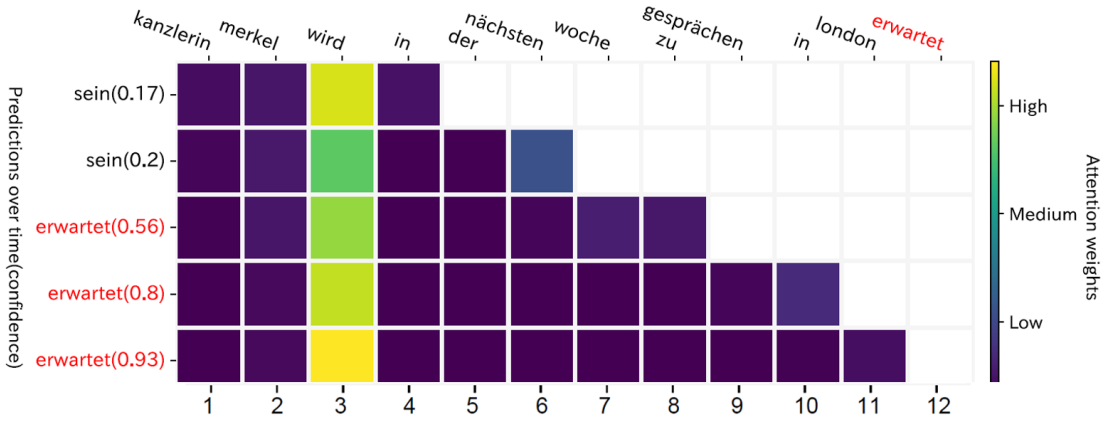


Figure 6.4: Transition of attention weights of a word-based attention model on a German sentence. The model successfully captures the passive voice in the sentence where "wird erwartet" is often translated together as "is expected". Full translation of the example is: Chancellor Merkel is expected to speak in London next week.

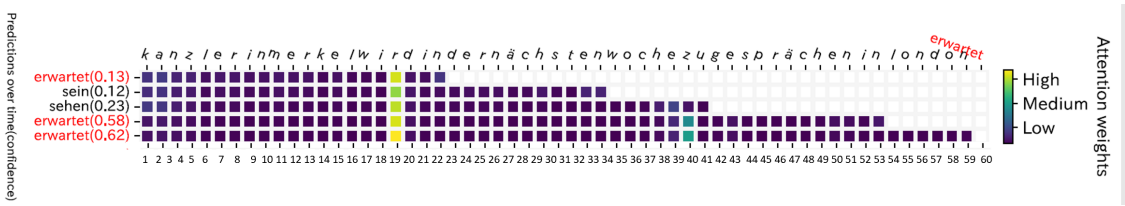


Figure 6.5: Transition of attention weights of a character-based attention model on a German sentence. Compared to Figure 6.4, the character-based model also makes the correct prediction and displays close attention on letter "i" in the word "wird", which is consistent to the word-based model. However, the heatmap is already extremely long for this short sentence and could be hard to interpret if without reference to the word-level heatmap.

dicting the normalized verb, we show in Section 5.5 that ANVIIL is also capable of predicting the tense of sentence-final verbs incrementally based on limited information of the preverbs.

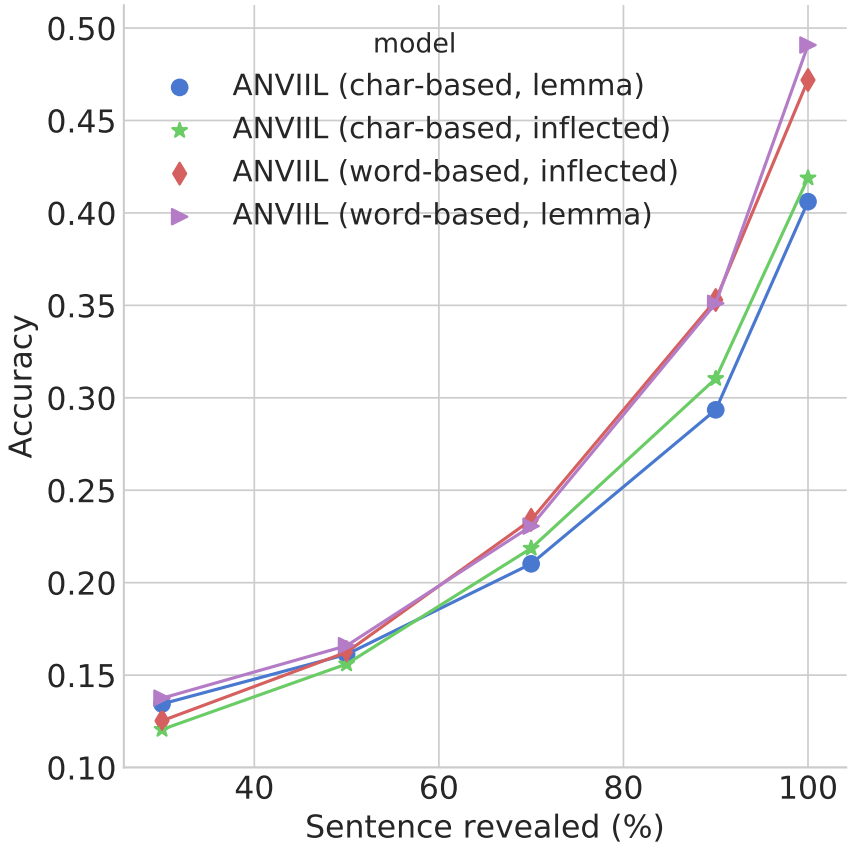


Figure 6.6: Accuracy of predicting inflected and lemmatized final-verbs in German are negligible.

6.2.3 Incomplete Sentences versus Complete Sentences

Existing neural attention models for classification and SMT are commonly trained on complete input sequences [13, 14, 15]. As mentioned in [11], long-distance sentence-final verb prediction in SOV languages and classification on incomplete sequences remains to be resolved.

Training on incomplete sentences has various advantages. First, as not every sentence is verb-final, samples that can be used for training and testing are limited even on

a very large corpus. For example, only 2.8 million sentences in the Wortschatz Leipzig news corpus from 2011 to 2015 (seven million sentences in total) are verb final, and only the sentences in them that satisfy the length and verb frequent constraints can be further utilized. As each preverb is truncated into multiple subsentences of different lengths, and each subsentence is taken as an independent sample, it allows us to train and test on more samples and reduces sparsity of the verbs. Second, in an incremental setting, future information is expected to be unavailable regarding to the current time step, which often precludes the chance of using BiRNNs on the complete sentences [11]. However, its practical to apply BiRNNs on incomplete sentences where all information till current are revealed, which often leads to better encoding by summarizing not only preceding but also following context of each word [15, 28].

Chapter 7: Conclusion

We present ANVIIL, a neural model for incremental verb prediction using BiGRU with self-attention. It outperforms existing models in predicting the most frequent sentence-final verbs in both Japanese and German. As we predict the verbs incrementally, our method can be directly applied to real-time sequential classification or prediction tasks. SMT systems for SOV to SVO simultaneous MT can also benefit from our work to reduce translation latency. Larger datasets always help with predicting the sentence-final verbs, suggesting that larger corpora will further improve results.

7.1 Future Work

Incorporating syntax information In experiments conducted by Grissom II et al. [3], case markers play an important role in human verb prediction on Japanese. They then incorporate case markers as one important feature in their statistical prediction model. Therefore, including syntax information such as learning additional embeddings of POS tags and case markers may further improve the prediction accuracy.

A more flexible assessment on the predicted verbs We now predict the sentence-final verb by minimizing the multi-class cross entropy loss. The predicted verb, which has the highest conditional probability given an input, is considered as incorrect

if it is not an exact match. In empirical studies, Jörg [4] divides successful anticipation of verbs into two categories: general and exact. A more natural and flexible assessment therefore should regard "near misses" prediction (such as "give" versus "provide") as a more general, but still successful prediction. This requires an improvement on the loss function which takes distance between the prediction and the target into consideration.

Simultaneous tense prediction for Japanese verbs As stated in Section 5.5, while ANVIIL is able to predict the exact inflected verb for German, due to the sparsity of Japanese verbs, separate process are needed to predict normalized Japanese sentence-final verbs and the corresponding tense. As we consider same input sequences and train under same model infrastructure, predicting tense and verbs for Japanese at the same time should be practical to implement and can better serve SMT systems.

Simultaneous machine translation with verb prediction Verb prediction is closely related to latency reduction in SOV to SVO SMT systems. Grissom II et al. [9] used a simple verb-specific n-gram model for sentence-final verb prediction and learned when to trust the predictions in a SMT system with reinforcement learning. Similarly, Gu et al. [11] incorporate the idea of prediction into actions of the agent during translation, however, with training only on complete source sentences, prediction could fail under circumstances when the verb is unseen in the source sentence. Having trained on incomplete sentences and considered hundreds of sentence-final verbs, it is nature to incorporate ANVIIL into a SMT system in the future.

Robust verb prediction with noisy text/speech input We use preprocessed text as input in the current model, however, noise is unavoidable in real life data. Typical examples of noisy text inputs including misspelled words and characters, handwritten

texts and electronically recognized documents. For speech inputs, which simultaneous interpretation must deal with, if we want to convert speech into texts, noise can be introduced by Automatic Speech Recognition (ASR) systems, and are also commonly rely on the speaker and environment. Therefore, learning sentence representations from noisy inputs, and reducing noise with techniques like automatic spelling correction will help make robust verb prediction in real life applications.

Improve prediction with context We predict sentence-final verbs incrementally for each input sentence, independently from the others. We apply BiGRU and self-attention to capture the context within the sentence and make predictions based on the learned representation. In fact, input sentences can be related to each other, where specific topics can be inferred from the global context. For example, if applied in a speech delivered by diplomats, the verbs we want to predict are more likely to be formal phrasal verbs and tightly related to politics. Under these circumstances, the selection of verbs can be limited to a smaller range than considering all the verbs in vocabulary. Hence, prediction accuracy may be improved by not only learning context on a sentence level, but also taking document context into consideration.

Extended prediction and online classification We show that larger datasets always help with predicting the sentence-final verbs, thus more robust prediction can be made with a larger corpora on a wider selection of verbs. As we predict the verbs incrementally, our method can be directly applied to solve real-time sequential classification or prediction problems, such as online sentiment analysis and word prediction. While next word prediction or word completion tasks have been studied to reduce keyboard stokes in input methods, long-distance prediction such as verb prediction can further help with

assistive writing and translation.

Bibliography

[1] Christopher Olah. Understanding lstm networks, 2015. URL http://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-chain.png. [Online; accessed June 20, 2018].

[2] Christopher Olah. Understanding lstm networks, 2015. URL http://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-var-GRU.png. [Online; accessed June 20, 2018].

[3] Alvin Grissom II, Naho Orita, and Jordan Boyd-Graber. Incremental prediction of sentence-final verbs: Humans versus machines. In *Conference on Computational Natural Language Learning*, pages 95–104, 2016.

[4] Udo Jörg. Bridging the gap: Verb anticipation in german-english simultaneous interpreting. In M. Snell-Hornby, Z. Jettmarová, and K. Kaindl, editors, *Translation as Intercultural Communication: Selected Papers from the EST Congress, Prague 1995*. 1997.

[5] He He, Jordan Boyd-Graber, and Hal Daumé III. Interpretese vs. translationese: The uniqueness of human strategies in simultaneous interpretation. In *North American Association for Computational Linguistics*, 2016.

[6] Lorenzo Bevilacqua. The position of the verb in Germanic languages and simultaneous interpretation. *The Interpreters' Newsletter*, 14:1–31, 2009.

[7] G.V. Chernov, R. Setton, and A. Hild. *Inference and Anticipation in Simultaneous Interpreting: A Probability-prediction Model*. Benjamins translation library. J. Benjamins Publishing Company, 2004. ISBN 9789027216632. URL https://books.google.com/books?id=cRbUqPeg9AcC.

[8] Shigeki Matsubara, Keiichi Iwashima, Nobuo Kawaguchi, Katsuhiko Toyama, and Yasuyoshi Inagaki. Simultaneous Japanese-English interpretation based on early predition of English verb. In *Symposium on Natural Language Processing*, 2000.

[9] Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. Don't until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Empirical Methods in Natural Language Processing*, 2014.

[10] Yoav Goldberg. *Neural Network Methods for Natural Language Processing*. Synthesis Lectures on Human Language Technologies. 2017.

[11] Jiatao Gu, Graham Neubig, Kyunghyun Cho, and year = 2017 volume = 1 pages = 1053–1062 booktitle = Long Papers - Continued publisher = Association for Computational Linguistics (ACL) Li, Victor O.K. Learning to translate in real-time with neural machine translation.

[12] Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. Syntax-based simultaneous translation through prediction of unseen syntactic constituents. *Proceedings of the Association for Computational Linguistics*, June 2015.

[13] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. Hierarchical attention networks for document classification. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2016.

[14] Sheng-syun Shen and Hung-yi Lee. Neural attention models for sequence classification: Analysis and application to key term extraction and dialogue act detection. In *Conference of the International Speech Communication Association*, 2016.

[15] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv e-prints*, 2014.

[16] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 1st edition, 2000.

[17] Ronald Rosenfeld. Two decades of statistical language modeling: Where do we go from here. In *Proceedings of the IEEE*, page 2000, 2000.

[18] Stanley F. Chen, Stanley F. Chen, Joshua Goodman, and Joshua Goodman. An empirical study of smoothing techniques for language modeling. Technical report, Harvard University, 1998.

[19] Zhengzheng Xing, Jian Pei, and Eamonn Keogh. A brief survey on sequence classification. *SIGKDD Explor. Newsl.*, 12(1):40–48, 2010. URL `http://doi.acm.org/10.1145/1882471.1882478`.

[20] Alvin Castillo Grissom II. *Incremental Prediction and Decision-Making for Simultaneous Machine Translation*. PhD thesis, University of Colorado at Boulder, 2017.

[21] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2267–2273, 2015.

[22] Zachary Chase Lipton. A critical review of recurrent neural networks for sequence learning. *CoRR*, abs/1506.00019, 2015. URL `http://arxiv.org/abs/1506.00019`.

[23] Fuchun Peng, Dale Schuurmans, and Shaojun Wang. Augmenting naive bayes classifiers with statistical language models. *Inf. Retr.*, 7(3-4):317–345, September 2004. ISSN 1386-4564.

[24] Lars Konieczny and Philipp Döring. Anticipation of clause-final heads. evidence from eye-tracking and srns. In *Proceedings of ICCS/ASCS*, pages 330–335, 01 2003.

[25] Graham Neubig. Neural machine translation and sequence-to-sequence models: A tutorial. *CoRR*, abs/1703.01619, 2017.

[26] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[27] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Conference of Empirical Methods in Natural Language Processing*, 2014.

[28] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681, November 1997.

[29] Volodymyr Mnih, Nicolas Heess, Alex Graves, and koray kavukcuoglu. Recurrent models of visual attention. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2204–2212. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5542-recurrent-models-of-visual-attention.pdf`.

[30] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. abs/1508.04025, 2015. URL `http://arxiv.org/abs/1508.04025`.

[31] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. In *Proceedings of the International Conference on Learning Representations*, 2017.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf`.

[33] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[34] Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. Building large monolingual dictionaries at the leipzig corpora collection: From 100 to 200 languages. In *International Language Resources and Evaluation*, 2012.

[35] Helmut Schmid. Improvements in part-of-speech tagging with an application to german. In *Proceedings of the ACL SIGDAT-Workshop*, 1995.

[36] Christos Baziotis, Nikos Athanasiou, Pinelopi Papalampidi, Athanasia Kolovou, Georgios Paraskevopoulos, Nikolaos Ellinas, and Alexandros Potamianos. NTUA-SLP at semeval-2018 task 3: Tracking ironic tweets using ensembles of word and character level attentive rnns. *CoRR*, abs/1804.06659, 2018. URL `http://arxiv.org/abs/1804.06659`.

[37] T. Kudo. Mecab : Yet another part-of-speech and morphological analyzer. *http://mecab.sourceforge.net/*, 2005.

[38] M. Wolfel, M. Kolss, F. Kraft, J. Niehues, M. Paulik, and A. Waibel. Simultaneous machine translation of german lectures into english: Investigating research challenges for the future. In *IEEE Spoken Language Technology Workshop*, 2008.

[39] Patrick Doetsch, Pavel Golik, and Hermann Ney. A comprehensive study of batch construction strategies for recurrent neural networks in mxnet. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2017.

[40] Makoto Morishita, Yusuke Oda, Graham Neubig, Koichiro Yoshino, Katsuhito Sudoh, and Satoshi Nakamura. An empirical study of mini-batch creation strategies for neural machine translation. In *The First Workshop on Neural Machine Translation*, 2017.

[41] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, 2015.

[42] Brian Joseph. What gives with es gibt? *American Journal of Germanic Linguistics and Literatures*, 12:243–265, 2000.