ABSTRACT

Title of dissertation: DETECTING AND RECOGNIZING HUMANS, OBJECTS, AND THEIR INTERACTIONS

Ankan Bansal
Doctor of Philosophy, 2020

Dissertation directed by: Professor Rama Chellappa
Department of Electrical and Computer Engineering

Scene understanding is a high-level vision task which involves not just localizing and recognizing objects and people but also inferring their layouts and interactions with each other. However, current systems for even atomic tasks like object detection suffer from several shortcomings. Most object detectors can only detect a limited number of object categories; face recognition systems are prone to make mistakes for faces in extreme poses or illuminations; and automated systems for detecting interactions between humans and objects perform poorly. We hypothesize that scene understanding can be improved by using additional semantic data from outside sources and intelligently and efficiently using the available data.

Given the fact that it is nearly impossible to collect labeled training data for thousands of object categories, we introduce the problem of zero-shot object detection (ZSD). Here, "zero-shot" means recognizing/detecting without using any visual data during training. We first present an approach for ZSD using semantic information encoded in word-vectors which are trained on a large text corpus. We

discuss some challenges associated with ZSD. The most important of these challenges is the definition of a "background" class in this setting. It is easy to define a "background" class in fully-supervised settings. However, it's not clear what constitutes a "background" ZSD. We present principled approaches for dealing with this challenge and evaluate our approaches on challenging sets of object classes, not restricting ourselves to similar and/or fine-grained categories as in prior works on zero-shot classification.

Next, we tackle the problem of detecting human-object interactions (HOIs). Here, again, it is impossible to collect labeled data for each type of possible interaction. We show that solutions for HOI detection can greatly benefit from semantic information. We present two approaches for solving this problem. In the first approach, we exploit functional similarities between objects to share knowledge between models for different classes. The main idea is that humans look similar while interacting with functionally similar objects. We show that, using this idea, even a simple model can achieve state-of-the-art results for HOI detection both in the supervised and zero-shot settings. Our second model uses semantic information in the form of spatial layout of a person and an object to detect their interactions. This model contains a layout module which primes the visual module to make the final prediction.

An automated scene understanding system should, further, be able to answer natural language questions posed by humans about a scene. We introduce the problem of Image-Set Visual Question Answering (ISVQA) as a generalization of existing tasks of Visual Question Answering (VQA) for still images, and video VQA.

We describe two large-scale datasets collected for this problem: one for indoor scenes and one for outdoor scenes. We provide a comprehensive analysis of the two datasets. We also adapt VQA models to design baselines for this task and demonstrate the difficulty of the problem.

Finally, we present new datasets for training face recognition systems. Using these datasets, we show that careful consideration of some critical questions before training can lead to significant improvements in face verification performance. We use some lessons from these experiments to train a face recognition system which can identify and verify faces accurately. We show that our model, trained with the recently introduced Crystal Loss, can achieve state-of-the-art performance for many challenging face recognition benchmarks like IJB-A, IJB-B, and IJB-C. We evaluate our system on the Disguised Faces in the Wild (DFW) dataset and show convincing first results.

DETECTING AND RECOGNIZING HUMANS, OBJECTS, AND THEIR INTERACTIONS

by

Ankan Bansal

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
2020

Advisory Committee:
Professor Rama Chellappa, Chair/Advisor
Professor Behtash Babadi
Professor Ramani Duraiswami
Professor Abhinav Shrivastava
Professor David Jacobs, Dean's Representative

For my parents and sister

Acknowledgments

I would like to express my profound gratitude to my advisor, Prof. Rama Chellappa, for the constant motivation, guidance, and support provided throughout last five years. He was always there to listen and discuss any ideas. His insights were helpful in dealing with roadblocks during my research. His advice has always been extremely precise and helpful. I am also thankful to Professor Abhinav Shrivastva for conducting detailed discussions and providing key insights about my work. I am also grateful to my other committee members, Professors Behtash Babadi, Ramani Duraiswami, and David Jacobs for their time and feedback.

I wish to thank Dr. Carlos D. Castillo, who was always ready to talk about my work and provided key insights. He is an amazing guide and taught me the ropes during the initial years of my PhD. I was very lucky to have him as my first mentor. Thanks are also due to Dr. Jun-Cheng Chen who devoted more time to research than everyone and still made time to discuss any problems I faced. I am also thankful to Dr. Swami Sankaranarayanan for helping me achieve a steep learning curve in the first year of my graduate studies.

I was fortunate to work with great mentors during my internships. These internships helped me understand the nitty-gritties of working on real-world problems in a fast-paced setting. I am deeply grateful to the mentorship provided by Dr. Ajay Divakaran and Dr. Karan Sikka at SRI International, Dr. Gaurav Sharma at NEC Labs, and Dr. Joaquin Zepeda and Dr. Yuting Zhang at Amazon AWS.

Immeasurable thanks are due to Ms. Janice Perrone for managing all of the

administrative requirements. She makes all of our lives much easier by constantly working hard. I would also like to take the opportunity to thank the administrative staff of ECE and UMIACS for their help throughout my graduate studies. In particular, I would like to thank Ms. Arlene Schenk, Ms. Vivian Lu, Ms. Maria Hoo, Ms. Melanie Prange, and Ms. Emily Irwin for all their help. I am also grateful to the nice people in the ECE Business Office and the janitorial staff in AVW and the Iribe Center for taking care of the big things so that I can focus on the tiny ones.

I want to take this opportunity to thank my colleagues and friends Saketh, Rajeev, Anirudh, Navaneeth, David, Anshul, Shlok, Prithvi, Sayantan, Upal for the great discussions on life, the universe, and everything. I am also grateful for the help and support provided by my lab-mates Andy, Hui, Arthita, Steve, Jingxiao, Boyu, Amit, Ilya, Josh, and Hongyu.

The path to this dissertation was a tough one. My friends were always there to take me through all the highs and lows. I could not have done this without their support. I am immensely grateful for having Shivangi, Heena, Sanchit, Abhinav, Sarthak, Kiran, Shashank, Lovely, and Saurabh as my friends.

I am thankful to C.G.P. Grey, Brady Haran, Destin Sandlin, John Green, Emily Graslie, Tom Scott, James Harkin, Andrew Hunter Murray, Anna Ptaszynski, Dan Schreiber, Roman Mars, Stephen Dubner, Guy Raz, and the Planet Money Team, for making me a bit smarter everyday.

My gratitude for my family is ineffable. But for their support and sacrifices, I wouldn't be writing this.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

Scene understanding is the process of reasoning about the humans, objects, and their spatial, functional, and semantic relationships. This involves localizing and recognizing all humans, and objects in a scene and understanding their interactions. Scene understanding will be an essential capability for automated computer vision systems.

Humans develop an instinctive ability to understand a given scene. A glimpse at the scene in Figure 1.1 is enough to inform us that Jack Nicholson is typing on a type-writer while sitting on a chair in the hall at the Overlook Hotel. Also, there is a book, and a table lamp on the table. Notice all the processes involved in making these deductions. We localize and may recognize the person, detect all the objects, recognize the place, and infer relationships among all of these entities. We achieve this by using common-sense reasoning and knowledge about the properties, affordances, and physics of objects.

Such an ability to understand the scenes completely has not been replicated in automated systems. With the huge success of deep learning, computers now have the ability to perform atomic tasks like detecting a limited set of object categories, and recognize people in relatively easy circumstances. However, computer vision

Figure 1.1: Humans can instantaneously understand a scene. We utilize our knowledge about the properties and affordances of objects to reason about interactions between different objects and people.

systems have a long way to go before they are able to understand relationships between objects, or recognize humans in extreme poses and illuminations, or detect objects for which no training data is available.

Lack of training data is a major hurdle in advancing several parts of scene understanding. The most popular object detection systems [83, 134, 175] are usually trained and evaluated on the MSCOCO dataset [130] which contains only 80 object categories. These systems require labeled training data for all categories. The world contains thousands of object classes. It is prohibitively expensive to collect labeled images/videos for all of them. Similarly, humans can interact with each of these objects in several different ways. This makes the problem of recognizing interactions between humans and objects combinatorial in nature. Collecting labeled data for every type of interaction is nearly impossible.

This dissertation is a step towards solving some of these problems for holis-

tic scene understanding. The main hypothesis is that scene understanding can be improved by using additional semantic data from outside source and intelligently and efficiently using the available data. Such semantic information can provide additional supervision for computer vision systems. This will eventually enable them to learn without any labeled data. Also, carefully collecting and merging available data can significantly improve the performance of different sub-systems.

Here is an overview of the contents of this dissertation.

We start by providing some background to the ideas introduced in this dissertation. In Chapter 2, we briefly discuss some prior works on object detection, visual relationship detection, and face recognition and detection. We discuss both hand-designed and CNN-based object detectors including Viola-Jones, DPM, R-CNN, Faster R-CNN etc. We also explore several recent advances in CNN-based face recognition.

We introduce and tackle the problem of Zero-Shot Object Detection (ZSD) in Chapter 3. In this context, the term "zero-shot" means that the models do not use any labeled visual data for some classes while training and are still able to infer those classes. In particular, ZSD means that visual supervised data is available only for a few object classes. The trained model is able to recognize previously unseen classes. We work with a challenging set of object classes, not restricting ourselves to similar and/or fine-grained categories as in prior works on zero-shot classification. We present a principled approach by first adapting visual-semantic embeddings for zero-shot detection. We then discuss the problems associated with

selecting a background class and motivate two background-aware approaches for learning robust detectors. One of these models uses a fixed background class and the other is based on iterative latent assignments. We also outline the challenge associated with using a limited number of training classes and propose a solution based on dense sampling of the semantic label space using auxiliary data with a large number of categories. We propose novel splits of two standard detection datasets – MSCOCO and VisualGenome, and present extensive empirical results in both the traditional and generalized zero-shot settings to highlight the benefits of the proposed methods. We provide useful insights into the algorithm and conclude by posing some open questions to encourage further research.

The next two chapters are dedicated to detecting human-object interactions. In Chapter 4 we present an approach for detecting human-object interactions (HOIs) in images, based on the idea that humans interact with functionally similar objects in a similar manner. The proposed model is simple and uses the visual features of the human, relative spatial orientation of the human and the object, and the knowledge that functionally similar objects take part in similar interactions with humans. We provide extensive experimental validation for our approach and demonstrate state-of-the-art results for HOI detection. On the HICO-Det dataset our method achieves a gain of over 2.5% absolute points in mean average precision (mAP) over recent works. We also show that our approach leads to significant performance gains for zero-shot HOI detection in the seen object setting. We further demonstrate that using a generic object detector, our model can generalize to interactions involving previously unseen objects.

The relative spatial layout of a human and an object is an important cue for determining how they interact. However, until now, spatial layout has been used just as side-information for detecting human-object interactions (HOIs). In Chapter 5, we present a method for exploiting this spatial layout information for detecting HOIs in images. The proposed method consists of a layout module which primes a visual module to predict the type of interaction between a human and an object. The visual and layout modules share information through lateral connections at several stages. The model uses predictions from the layout module as a prior to the visual module and the prediction from the visual module is given as the final output. It also incorporates semantic information about the object using word2vec vectors. The proposed model reaches an mAP of 24.79% for HICO-Det dataset which is about 5.4% absolute points higher than the current state-of-the-art. For zero-shot HOI detection, the proposed approach performs about three times better than state-of-the-art.

Holistic scene understanding involves being able to answer human questions about a scene. In Chapter 6, we introduce the task of Image-Set Visual Question Answering (ISVQA), which generalizes the commonly studied single-image VQA problem to multi-image settings. Taking a natural language question and a set of images as input, it aims to answer the question based on the content of the images. The questions can be about objects and relationships in one or more images or about the entire scene depicted by the image set. To enable research in this new topic, we introduce two ISVQA datasets – indoor and outdoor scenes. They simulate the real-world scenarios of indoor image collections and multiple car-mounted cameras,

respectively. The indoor-scene dataset contains 91,479 human-annotated questions for 48,138 image sets, and the outdoor-scene dataset has 49,617 questions for 12,746 image sets. We analyze the properties of the two datasets, including question and answers distributions, types of questions, biases in dataset, and question-image dependencies. By adapting existing VQA methods, we also build new baseline models to investigate new research challenges in ISVQA. These challenges necessitate the development of new methods to understand the image-level relationships and perform cross-image reasoning for ISVQA tasks.

Next, in Chapters 7 and 8, we focus on face recognition. We introduce two large-scale annotated datasets which can help advance research in this area. While the research community appears to have developed a consensus on the methods of acquiring annotated data, design and training of CNNs, many questions still remain to be answered. We use our collected datasets to study the following questions that are critical to face recognition research: (i) Can we train on still images and expect the systems to work on videos? (ii) Are deeper datasets better than wider datasets? (iii) Does adding label noise lead to improvement in performance of deep networks? (iv) Is alignment needed for face recognition? Further, we use these datasets and the insights from our experiments to train two different networks to tackle the problem of recognizing disguised faces. Unconstrained face verification is a challenging problem owing to variations in pose, illumination, resolution of image, age, etc. This problem becomes even more complex when the subjects are actively trying to deceive face verification systems by wearing a disguise. The problem under consideration here is to identify subjects under disguise and reject impostors trying to look like a

subject of interest. We fuse features obtained from the two networks and show that the resulting features are effective for discriminating between disguised faces and impostors in the wild. We present results on the recently introduced Disguised Faces in the Wild challenge dataset.

Finally, we summarize our work and discuss some directions for further work to understanding the use of semantic information for scene understanding Chapter 9. In particular, we discuss video scene understanding by exploiting objects in a video and the actions they evoke. We also discuss some work planned for action recognition from videos using such semantic reasoning.

Chapter 2: Background

This chapter describes some of the theory and background related to the topics in this dissertation. We start by describing early and recent methods for object detection. Our work on ZSD, described in Chapter 3 extends these fully-supervised methods to the zero-shot learning setting. We then describe the general problem of visual relationship detection (VRD) and discuss some existing strategies for solving this problem. Human-object interaction detection (Chapters 4 and 5) is a subset of VRD where the subject is always a human. Next, we describe some recent advances in VQA to provide a background for our work on image-set VQA (Chapter 6). Finally, we discuss CNN-based face recognition to provide a context for our work on face verification and identification in Chapters 7 and 8.

2.1 Object Detection

Among the earliest methods for object detection, [205] uses hand-designed Haar-like features as representations of regions in an image. The **Viola-Jones** detector [205] combines increasingly complex classifiers into a cascade which improves detection performance at each step. Such a cascade is able to discard background regions early and devotes more computational resources to more promising regions. The authors

also proposed the idea of an integral image which enables the computation of these Haar-like features at each sliding window location a constant-time operation. The Viola-Jones detector also uses Adaboost to select a small number of features out of a large set of features obtained from the sliding windows. Constant-time computation of features, early rejection of background regions using a cascade of detectors, and high quality feature selection enable the Viola-Jones detector to detect faces in real-time and with high accuracy [205, 206].

Soon after Viola-Jones, Dalal and Triggs [39] proposed grids of **Histograms of Oriented Gradients** (HOGs) as a feature representation for detecting pedestrians in images. At each sliding window the authors propose to extract HOG feature vectors over a grid of overlapping blocks. These feature vectors are collected into a descriptor for the window and a linear SVM is used to classify this feature as human vs non-human. The authors also describe several tricks and techniques for improving the quality of HOG descriptors. These include gamma and color normalization, weighted voting into spatial and orientation cells, and contrast normalization in overlapping descriptor blocks.

The two methods described above work well for rigid objects. However, these methods give low performance for highly variable and deformable object classes. To overcome these issues [53] proposed **Discriminately Trained Part-Based Models** for detecting deformable objects. DPM extends the Dalal and Triggs HOG detector by considering a star-structure for parts and computing the corresponding part filters and deformation models along with a root filter. DPM looks at the problem of object detection in a bottom-up manner. An object can be detecting

by detecting parts of the object and combining those parts. The part filters are automatically learned in a weakly supervised setting.

DPM paved the way for deep CNN-based object detectors. These object detectors have become ubiquitous. CNN-based object detectors can be typically divided into two types: two-stage and one-stage. We describe some of the most popular models of both types next.

Two-Stage CNN Object Detectors

Two-stage detectors are called so because they consist of a region proposal stage and an object recognition stage. The region proposal stage produces a set of candidate object bounding boxes. The second stage is responsible for classifying each candidate region as either belonging to one of the object categories or to "background". R-CNN [67] can be considered as the first two-stage object detection model. Every other subsequent model has been derived from R-CNN. We start by briefly describing the R-CNN model and then discuss more recent developments.

Region-based convlutional networks (R-CNNs) [67] first extracts about 2,000 region proposals from an image using selective search [204]. Selective search is a multiple segmentation approach for object detection. Each segmented region from Selective Search is considered as a candidate for the next stage of object recognition. R-CNN crops each region proposal independently and warps it into a fixed shape. The authors then use a CNN to extract features for each of these independent cropped regions. Finally, the features are classified into a fixed set of categories using class-specific linear SVMs.

R-CNNs and other two-stage object detectors also use class-specific bounding

box regressors to output a new box after a region proposal is classified with class-specific SVMs. Given a pair of proposal bounding box $P = \{P_x, P_y, P_w, P_h\}$, the goal of bounding box regression is to produce a new box $\hat{G}$ which has a better overlap with the ground-truth box G. Here, (P_x, P_y) is the center coordinate, and P_w and P_h are the width and height of the proposal respectively. The ground-truth box is represented similarly. The final prediction is obtained using the following equations:

$$\hat{G}_x = P_w d_x(P) + P_x \tag{2.1}$$

$$\hat{G}_y = P_h d_y(P) + P_y \tag{2.2}$$

$$\hat{G}_w = P_w \exp(d_w(P)) \tag{2.3}$$

$$\hat{G}_h = P_h \exp(d_h(P)) \tag{2.4}$$

where $(d_x(P), d_y(P))$ represent the scale-invariant translation of the center of P, and $d_w(P)$ and $d_h(P)$ specify the log-space translations of the width and height. All of the $d_*(P)$ are modeled as linear transformations of the feature representation of P from the CNN. The weights of the linear transformations are learned using a mean-squared error with the targets:

$$t_x = (G_x - P_x)/P_w \tag{2.5}$$

$$t_y = (G_y - P_y)/P_h \tag{2.6}$$

$$t_w = \log(G_w/P_w) \tag{2.7}$$

$$t_h = \log(G_h/P_h) \tag{2.8}$$

A major shortcoming of R-CNN is that it is extremely slow due to the large number of object proposals and the fact that each object proposal has to be encoded separately by passing it through a deep network. **Fast R-CNN** [66] attempts to overcome this issue by using a fixed convolutional map for the whole image. The authors introduce the RoI pooling layer which pools the feature for each proposed region of interest into a fixed spatial dimension. This enables the re-use of the features obtained from the whole image. Each RoI is mapped to the fixed convolutional map for the image and the corresponding location is pooled to a fixed spatial extent. Fully-connected layers are applied to the RoI-pooled feature to obtain a fixed dimension feature vector for each RoI. This feature vector is finally passed to the two heads of the network: classification and bounding box regression. The whole network is trained using the following multi-task loss:

$$L(p, u, d^u, t) = L_{\text{cls}}(p, u) + \lambda L_{\text{loc}}(d^u, t) \tag{2.9}$$

where p is the probability distribution estimated by the classification head, u

is the ground-truth class, $d^u = \{d^u_x(P), d^u_y(P), d^u_w(P), d^u_h(P)\}$ are the translations of P for the class u as described above, L_{cls} is the cross-entropy loss, L_{loc} is a smoothed L_1 loss with targets $t = \{t_x, t_y, t_w, t_h\}$ described in Equation 2.8, and λ is the scaling factor.

Even though RoI-pooling helped in making object detection with Fast R-CNN slightly more efficient than R-CNN, the speed of the model was still limited by the time taken to generate high quality proposals. Implementations of selective search are slow. The authors of **Faster-RCNN** [175] proposed region-proposal network (RPN) to overcome this limitation. The idea behind RPN is that the "objectness" of a region can be predicted by a neural network. The authors incorporated a small CNN which can take a convolutional feature map and output a small number of region proposals which are highly likely to contain objects. This region-proposal network can share most of the weights in the trunk of the network with the Fast R-CNN. Such a process eliminates the need for a separate bounding box proposal step and can be trained end-to-end.

Mask R-CNN [83] further improves the speed and accuracy of Faster R-CNN by jointly learning multiple tasks like detection and segmentation. The authors showed that the Mask-RCNN framework can be further extended to tasks like human pose estimation.

One-Stage CNN Object Detectors

In one-stage object detectors, the most important difference from two-stage object detectors is the lack of an explicit region proposal step. Most of the methods that we discuss here contain implicit regions of interest instead. A major advantage of one-

stage detectors is the high speed compared to the two-stage detectors. For example, SSD [134] can run at upto 60 fps and YOLO [173] at around 45 fps compared to about 7 fps for Faster R-CNN.

Redmon *et al.* proposed **YOLO** [173] as a unified framework for object detection. YOLO frames object detection as a regression problem instead of a classification problem as typically done in two-stage detectors. After the last convolutional layer in the network, YOLO consists of two fully-connected layers: the first one takes the convolutional feature map to a fixed length feature vector, and the second projects the feature vector into a $S \times S \times (5B + C)$ tensor of predictions. Here S is the grid-size, B is the number of bounding boxes predicted for each grid cell, and C is the number of object classes.

In contrast to YOLO, the **Single-Shot Multibox Detector** (SSD) [134] considers a set of default boxes at different scales and aspect ratios at each feature map location. The model obtains such default boxes at various levels in the network to enable detection of objects at different resolutions. Object detections are made by classifying each default box and adjusting the bounding box. Similar to Faster R-CNN, the network is trained with a multi-task loss comprising of a cross-entropy term for classification and smoothed L_1 loss for bounding box regression.

More recently, Lin *et al.* [129] introduced Focal Loss which deals with the issue of class and easy-hard imbalance. Focal loss is a modification of cross-entropy loss and uses a focusing parameter to balance the imbalance between easy and hard samples. The authors also proposed a one-stage model, called RetinaNet, which is a modification of Feature Pyramid Network [128] and showed that such a one-stage

model can achieve higher speeds and accuracies that two-stage object detectors.

2.2 Visual Relationship Detection

A visual relationship is typically represented as the triplet ⟨`subject, predicate, object`⟩, i.e., visual relationships can be considered as a pair of objects linked by a predicate. The task of VRD requires correct localization of the two objects involved and the correct identification of the predicate. Due to the combinatorial nature of the problem - the number of possible relationships is the product of the square of the number of possible objects and the number of possible predicates - VRD is a much more challenging problem than object detection. For the same reasons, VRD cannot simply be considered as just a combination of atomic tasks of object detection and predicate recognition. In fact, relationships can provide important cues for detecting objects based on proximities and relationships with other objects.

In [99], the authors presented a framework for semantic image retrieval using scene graphs. Such scene graphs are constructed using the objects present in the image (e.g. person, car, building, bench), their attributes (e.g. car is red), and visual relationships between objects (e.g. ⟨`person, sitting, bench`*rangle*). The proposed model uses scene graphs as queries for retrieving similar images. VRD is an essential component for such a system for inferring the relationships between each pair of objects.

Long tail distribution of visual relationship categories is a highly apparent challenge for VRD. With increasing numbers of objects and predicates, the possible

number of interactions between objects increases rapidly. This hinders the collection of large amounts of labeled data for all relationship categories. Therefore, there is a need for models which can exploit similarities between objects and predicates to generalize from labeled samples for one category to rarer categories. For example, having seen a person riding a horse, and a camel walking on the ground, it should not be difficult to recognize a human riding a camel. With their VRD with Language Priors model [137], Lu *et al.* attempt to do exactly this. They propose to train visual models for objects and predicates separately. These models can be combined together to detect relationships between objects. The authors propose to use language priors in the form of word embeddings to model the likelihood of a predicted relationship.

Human-Object Interaction detection is a special case of Visual Relationship Detection. In HOI detection, the subject is a human. This reduces the complexity of VRD to only those relationships which involve a human. However, HOI detection is still an immensely challenging problem due to the varied poses taken by humans and the varied uses that they find for different objects. In this dissertation, we target the problem of HOI detection and propose two approaches - one based on the idea that humans tend to interact with functionally similar objects in a similar manner; and the other based on insight that the relative spatial layout of the subject and the object can provide enough information to form a prior prediction quickly. We discuss some background for HOI detection in Chapters 4 and 5.

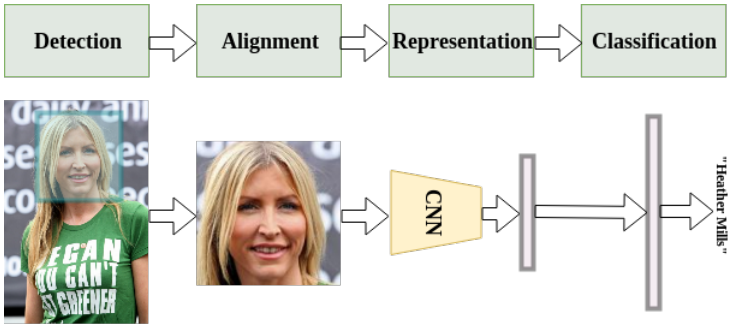


Figure 2.1: Standard approach for training a CNN for face verification and identification.

2.3 Face Recognition and Detection

Automatic face recognition is the problem of identifying a person from an image or a video. The problem of face recognition can be divided into face identification and face verification. The standard approach for training a CNN for solving these problems include four steps: face detection, alignment, representation, and classification (Figure 2.1). Identification is the problem of assigning an identity to an image from a list of identities. From another perspective, this can be considered as trying to retrieve the best matching face from a gallery for a given probe image. On the other hand, face verification involves verifying whether two face images are of the same person. This is usually performed by computing the similarity between feature representations of the two faces. Both identification and verification have benefited immensely from developments in deep learning algorithms and more advanced CNN architectures.

2.3.1 Datasets for Face Recognition

A face recognition system starts with detecting faces, then localizes landmarks which are used to align the faces to canonical views, and then classifies the detected faces. All three parts of the system require different level of information and data types. In this section, we explore some recently released public datasets targeted these.

In the wild face recognition at a large scale essentially started with the release of the Labeled Faces in the Wild (LFW) dataset [91]. Recent years have seen several large datasets being released to help the training of deep networks and to provide stronger benchmarks. Some examples of such include CelebA [136], CASIA-WebFace [237], MS-Celeb-1M [73], VGGFace [156], VGGFace2 [26], DFW [115] etc. However, these are still constrained because they only contain still images of mainly celebrities. Such photos are typically frontal and taken under good lighting. However, evaluation datasets like IJB-A [107], IJB-B [216], IJB-C [144], IJB-S [102], and Megaface [105] contain videos and images in varied conditions. To fill this gap, several video datasets have been proposed over the years. Among these, YouTube Faces (YTF) [217] is currently the largest publicly available annotated video dataset.

The most popular and the largest dataset for training and evaluating face detection models is the WIDER FACE dataset [231]. Another standard benchmark is FDDB [95]. The IARPA JANUS Benchmark datasets [107, 144, 216] also contain a large number of face annotations for evaluating face detection and recognition in completely unconstrained settings. Due to the difficulty in labeling and verifying facial keypoints in images, there are only a few large-scale public datasets available

which include such annotations. These include: Annotated Face in the Wild (AFW) [253], 300 faces-in-the-wild dataset [179], Labeled Face Parts in-the-wild (LFPW) [23], and Annotated Facial Landmarks in the Wild (AFLW) [110].

In addition to these, there are some 3D datasets [49], age datasets [122,176], attribute datasets [81,136], and expression datasets [136].

2.3.2 Face Detection

Face detection is the process of finding a bounding box for each face in an image. This is often the first step in any face recognition or tracking system. Counting the number of people in a crowded scene [22, 190] can also benefit from robust face and head detection. Large real-world datasets like [231] and deep CNN-based representations have led to significant improvements in face detection performance. Most of the popular face detection methods have been adapted from general object detectors and can be classified as either proposal-based or single-stage detectors.

Proposal-based object detection methods start with a class-agnostic object proposal generator like selective search [204], edge-boxes [257], or a region-proposal network (RPN) [175]. These proposals are then classified into object classes by a CNN. Proposal-based face detectors follow a similar approach and generate face proposals which are then classified as face vs non-face by a CNN. Examples of such face detectors include All-in-One Face [172], Hyperface [171], Finding Tiny Faces [89], and Supervised Transformer Network [33].

Unlike proposal-based detectors, single-stage object detectors do not contain

an explicit proposal generation step. Such detectors typically include a single-pass through a CNN and processing multi-scale image pyramid or multiple layers of a CNN. Single-shot multibox detector (SSD) [134] and YOLO [173,174] are examples of recent single stage object detectors. Several recent face detectors adapt these methods. These include DPSSD [168], SSH [149], CNN Cascade [123], ScaleFace [232], S^3FD [242].

After a face has been detected, the step in most face recognition pipelines in facial landmark detection and face alignment. Landmarks determine the most discriminative locations on a face. We refer the reader to the brief overview in [168] and a comprehensive review in [211] for a better coverage of the topic.

2.3.3 Loss Functions

The loss function is an important factor in determining the performance of deep networks. Most face recognition networks are trained to perform a $C-$way classification of faces with the hope that the learned features can be used as discriminative representations. Many existing works use the standard cross-entropy loss with softmax for training face recognition networks. Variants of the cross-entropy loss aim to address issues like preference for high quality images, early saturation, lack of margin between intra and inter-class samples etc. Some methods instead focus on directly optimizing the features for face verification. Metric learning approaches optimize the features to reduce intra-class separation and increase inter-class separation.

We start with a description of the standard softmax based cross-entropy loss.

Suppose there are M training samples in a batch. Let, $\mathbf{x}_i$ be the i^{th} face image in the batch with the label y_i and $f(\mathbf{x}_i)$ be the feature representation of the face. The feature representation is typically a deep CNN. The feature vectors are projected into logits using weights W and bias b. Then, the softmax loss is given by:

$$\mathcal{L}_{Softmax} = -\frac{1}{M}\sum_{i=1}^{M}\log\frac{e^{W_{y_i}^T f(\mathbf{x}_i)+b_{y_i}}}{\sum_{j=1}^{C} e^{W_j^T f(\mathbf{x}_i)+b_j}} \tag{2.10}$$

where, C is the total number of classes, W_j is the j^{th} column of the weight matrix W and b_j is the corresponding bias. Note that the bias term in Equation 2.10 can be absorbed into the weights by appending 1 to $f(\mathbf{x}_i)$. Now, since $\mathbf{a}^T\mathbf{b} = \|a\|\|b\|\cos(\theta)$, where θ is the angle between $\mathbf{a}$ and $\mathbf{b}$, the equation above can be re-written as:

$$\mathcal{L}_{Softmax} = -\frac{1}{M}\sum_{i=1}^{M}\log\frac{e^{\|W_{y_i}\|\|f(\mathbf{x}_i)\|\cos(\theta_{y_i})}}{\sum_{j=1}^{C} e^{\|W_j\|\|f(\mathbf{x}_i)\|\cos(\theta_j)}} \tag{2.11}$$

At test time, a probe face $\mathbf{x}_p$ is compared to a face in the gallery, $\mathbf{x}_g$ using cosine similarity:

$$s = \frac{f(\mathbf{x}_p)^T f(\mathbf{x}_g)^T}{\|f(\mathbf{x}_p)\|_2 \|f(\mathbf{x}_g)\|_2} \tag{2.12}$$

A-Softmax [135] incorporates an angular margin to the softmax formulation. This is based on the idea that at test time, we usually want dissimilar features to be angularly separated (since our distance metric is cosine distance). A-Softmax starts by normalizing the weight vectors $\|W_j\| = 1,\ \forall j$. Let, $\|f(\mathbf{x}_i)\| = s$, then the

A-Softmax loss is give as:

$$\mathcal{L}_{SphereFace} = \frac{-1}{M}\sum_{i=1}^{M}\log\frac{e^{s\cos(m\theta_{y_i,i})}}{e^{s\cos(m\theta_{y_i,i})}+\sum_{j\neq y_i}e^{s\cos(\theta_{j,i})}} \tag{2.13}$$

where m is the size of the margin and $\theta_{y_i,i}$ is in the range $\left[0, \frac{\pi}{m}\right]$. However, training a CNN under this constraint is difficult. Therefore, the authors in [135] propose to generalize $\cos(\theta_{y_i,i})$ to a monotonic angle function $\psi(\theta_{y_i,i})$ which equals $\cos(\theta_{y_i,i})$ in $\left[0, \frac{\pi}{m}\right]$. So, A-softmax can be written as:

$$\mathcal{L}_{SphereFace} = \frac{-1}{M}\sum_{i=1}^{M}\log\frac{e^{s\psi(\theta_{y_i,i})}}{e^{s\psi(\theta_{y_i,i})}+\sum_{j\neq y_i}e^{s\cos(\theta_{j,i})}} \tag{2.14}$$

where $\psi(\theta)$ is a piecewise function:

$$\psi(\theta) = (-1)^k\cos(m\theta) - 2k,\ \theta \in \left[\frac{k\pi}{m}, \frac{(k+1)\pi}{m}\right] \quad \text{and } k \in [0, m-1] \tag{2.15}$$

Large Margin Cosine Loss [210] uses an additive margin term instead of a multiplicative margin as used above. In addition to fixing $\|W_j\| = 1$ by L_2 normalization, the authors propose to fix $\|f(\mathbf{x}_i)\| = s$. This puts the learned features on a hypersphere were they need to be separable in the angular space. Fixing the norm of the features is a commonly used technique, e.g. [167]. Adding the margin in Equation 2.11 thus gives the formulation:

$$\mathcal{L}_{CosFace} = -\frac{1}{M}\sum_{i=1}^{M}\log\frac{e^{s(\cos(\theta_{y_i,i})-m)}}{e^{s(\cos(\theta_{y_i,i})-m)}+\sum_{j\neq y_i}e^{s\cos(\theta_{j,i})}} \tag{2.16}$$

The margin m and the feature scale s are inter-dependent.

Additive Angular Margin Loss [42] also starts by normalizing W_{y_i} and scaling the feature such that $\|f(\mathbf{x}_i)\| = s$. However, instead of directly adding an additive cosine margin as in Equation 2.16, [42] proposes to use an additive angular margin. This is again done with the aim of increasing inter-class discrepancy and intra-class compactness. The proposed loss can be written as:

$$\mathcal{L}_{ArcFace} = \frac{-1}{M}\sum_{i=1}^{M}\log\frac{e^{s(\cos(\theta_{y_i,i}+m))}}{e^{s(\cos(\theta_{y_i,i}+m))}+\sum_{j\neq y_i}e^{s\cos(\theta_{j,i})}} \tag{2.17}$$

where m is the additive angular margin. Additionally, the authors also propose a loss which combines SphereFace (Equation 2.13), CosFace (Equation 2.16), and the proposed ArcFace (Equation 2.17):

$$\mathcal{L}_{Combined} = -\frac{1}{M}\sum_{i=1}^{M}\log\left(\frac{e^{s(\cos(m_1\theta_{y_i,i}+m_2)-m_3)}}{e^{s(\cos(m_1\theta_{y_i,i}+m_2)-m_3)}+\sum_{j\neq y_i}e^{s\cos(\theta_{j,i})}}\right) \tag{2.18}$$

where m_1, m_2, and m_3 are the corresponding margins for SphereFace [135], ArcFace [42], and CosFace [210].

Several other loss functions have been proposed for training face recognition networks. However, space limitations do not allow a more detailed exposition of those methods. We refer the reader to the original papers for Noisy Softmax [32], Center Loss [215], Center Invariant Loss [219], Range Loss [243], Centralized Coordinate Learning [163], Ring Loss [248], Triplet Loss [184].

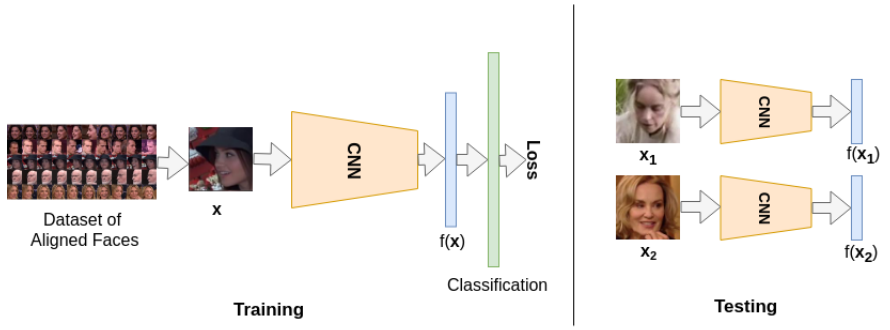


Figure 2.2: A face verification training and testing pipeline. A dataset of aligned faces is used to train a deep CNN with a classification loss. At test time, features are extracted from two faces and their similarity is computed to determine whether the two faces are of the same person.

2.3.4 Applications

In this section we describe some recent face recognition applications which utilize some of the techniques described above. We note that both face identification and verification can be formulated as the same problem. In identification, given a probe image, the goal is to find the closest image from a gallery. This is achieved by computing the similarities between the feature representation of the probe image and feature representations of the gallery images. The image with the highest similarity with the probe images is given as output. In verification, the aim is to determine if a given pair of images belong to the same person. This is also achieved by computing the similarity between the feature representations of the two images. The basic operation in both identification and verification is to extract a feature representation and compare with representations of the other image/images. We focus on face verification in this section. Similar methods can be used for face identification too.

A typical face verification training and testing pipeline is shown in Figure 2.2.

A training set of aligned faces is used to train a deep network for $C-$way classification. The layer before the classification layer is used to extract a feature representation for a face at test time. Representations from two faces are compared using a similarity metric.

DeepID [197] proposes to train a deep network on a large number of classes to obtain discriminative features which can be used for face verification. It extracts features from 60 face patches from different scales, different regions, and RGB or gray channels. Features for each patch and their flipped versions are extracted and concatenated into a 19,200 dimension feature. All neural networks are trained with softmax loss over a training dataset containing 10,177 identities.

DeepFace [199] uses explicit 3D modeling, starting from 2D keypoints, to apply a piecewise affine transformation for aligning faces. The aligned face is further warped to the image plane of a generic 3D face shape. After the alignment, DeepFace uses a nine layer deep network with 120 million parameters to learn the face representation. The network is trained using a dataset of four million images from over 4,000 identities. This network is trained with the standard softmax cross-entropy loss.

FaceNet [184] uses a triplet loss to directly optimize the embedding instead of using the surrogate task of $C-$way classification. The authors claim that this leads to greater representational efficiency and this feature embedding can improve face verification and clustering performance.

VGGFace [156] model uses a large dataset of over 2.6 million images from about 2,600 identities to train a CNN with softmax loss. The features obtained from

this network are embedded using a triplet loss similar to [132].

All-in-One Face [172] proposes a multi-task learning approach for face detection, keypoint detection, pose estimation, smile detection, gender classification, age estimation and face recognition. The network contains several heads which are responsible for learning different functionalities. The idea is that each modality will benefit from other modalities. The separate heads are trained with the corresponding losses and gradients from all heads are accumulated to train the trunk of the network. The face recognition/feature learning branch uses a standard softmax loss.

ArcFace [42] uses the Additive Angular Margin Loss and the large-scale, and clean MS1MV2 dataset to achieve state-of-the-art performance on several face recognition and verification benchmarks. The MS1MV2 dataset is a refined version of the MS-Celebl-1M dataset and contains about 5.8M faces for 85,000 identities. ArcFace uses the popular ResNet-100 network architecture.

Some other recent methods include [6, 28, 142, 143, 194, 209, 218, 225, 230].

Chapter 3: Zero-Shot Object Detection

Humans can effortlessly make a mental model of an object using only textual description, while machine recognition systems, until not very long ago, needed to be shown visual examples of every category of interest. Recently, some work has been done on *zero-shot* classification using textual descriptions [222], leveraging progress made on both visual representations [198] and semantic text embeddings [100, 146, 158]. In zero-shot classification, at training time visual examples are provided for some visual classes but during testing the model is expected to recognize instances of classes which were not seen, with the constraint that the new classes are semantically related to the training classes.

This problem is solved within the framework of transfer learning [57, 161], where visual models for seen classes are transferred to the unknown classes by exploiting semantic relationships between the two. For example, as shown in Figure 3.1, the semantic similarities between classes "hand" and "arm" are used to detect an instance of a related (unseen) class "shoulder". While such a setting has been used for object classification, object detection has remained mostly in the fully supervised setting as it is much more challenging. In comparison to object classification, which aims to predict the class label of an object in an image, object detection

aims at predicting bounding box locations for multiple objects in an image. While classification can rely heavily on contextual cues, e.g. vehicles are usually on roads, detection needs to exactly localize the object of interest and can potentially be degraded by contextual correlations [240]. Furthermore, object detection requires learning additional invariance to appearance, occlusion, viewpoint, aspect ratio etc. in order to precisely delineate a bounding box [86].

In the past few years, several CNN-based object detection methods have been proposed. Early methods [66, 67] started with an object proposal generation step and classified each object proposal as belonging to a class from a fixed set of categories. More recent methods either generate proposals inside a CNN [175], or have implicit regions directly in the image or feature maps [134, 173]. These methods achieved significant performance improvements on small datasets which contain tens to a few hundreds of object categories [47, 130]. However, the problem of detecting a large number of classes of objects has not received sufficient attention. This is mainly due to the lack of available annotated data as getting bounding box annotations for thousands of categories of objects is an expensive process. Scaling supervised detection to the level of classification (tens to hundreds of thousands of classes) is infeasible due to prohibitively large annotations costs. Recent works have tried to avoid such annotations, e.g. [174] proposed an object detection method that can detect several thousand object classes by using available (image-level) class annotations as weak supervision for object detection. Zero-shot learning has been shown to be effective in situations where there is a lack of annotated data [56, 58, 133, 155, 222, 226, 245, 246]. Most prior works on zero-shot learning have ad-

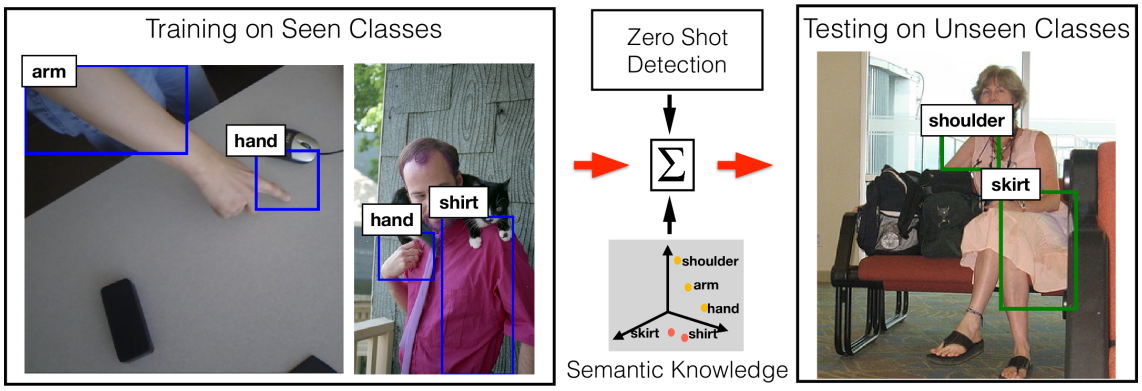


Figure 3.1: We highlight the task of zero-shot object detection where objects "arm", "hand", and "shirt" are observed (seen) during training, but "skirt", and "shoulder" are not. These unseen classes are localized by our approach that leverages semantic relationships between seen and unseen classes along with the proposed ZSD framework. The example has been generated by our model.

dressed the classification problem [24,29,46,55,94,108,117,118,154,164,166,193,221], using semantic word-embeddings [55,108] or attributes [56,118,125,245] as a bridge between seen and unseen classes.

In the present work, we introduce and study the challenging problem of *zero-shot detection* for diverse and general object categories. This problem is difficult owing to the multiple challenges involved with detection, as well as those with operating in a zero-shot setting. Compared to fully supervised object detection, zero-shot detection has many differences, notably the following. While in the fully supervised case a background class is added to better discriminate between objects (e.g. car, person) and background (e.g. sky, wall, road), the meaning of "background" is not clear for zero-shot detection, as it could involve both background "stuff" as well as objects from unannotated/unseen classes. This leads to non-trivial practical problems for zero-shot detection. We propose two approaches to address this problem: one using a fixed background class and the other using a large open vocabulary

for differentiating different background regions. We start with a standard zero-shot classification architecture [57] and adapt it for zero-shot object detection. This architecture is based on embedding both images and class labels into a common vector space. In order to include information from background regions, following supervised object detection, we first try to associate the background image regions into a single background class embedding. However, this method can be improved by using a latent assignment based alternating algorithm which associates the background boxes to potentially different classes belonging to a large open vocabulary. Since most object detection benchmark datasets usually have a few hundred classes, the label space can be sparsely populated. We show that dense sampling of the class label space by using additional data improves zero-shot detection. Along with these two enhancements, we provide qualitative and quantitative results to provide insights into the success as well as failure cases of the zero-shot detection algorithms, that point us to novel directions towards solving this challenging problem.

To summarize, our main contributions are: (i) we introduce the ZSD problem in real world settings and present a baseline method for ZSD that follows existing work on zero-shot image classification using multimodal semantic embeddings and fully supervised object detection; (ii) we discuss some challenges associated with incorporating information from background regions and propose two methods for training background-aware detectors; (iii) we examine the problem with sparse sampling of classes during training and propose a solution which densely samples training classes using additional data; and (iv) we provide extensive experimental and ablation studies in traditional and generalized zero-shot settings to highlight

the benefits and shortcomings of the proposed methods and provide useful insights which point to future research directions.

3.1 Related Work

Word embeddings. Word embeddings map words to a continuous vector representation by encoding semantic similarity between words. Such representations are trained by exploiting co-occurrences in words in large text corpuses [100, 146, 147, 158]. These word vectors perform well on tasks such as measuring semantic and syntactic similarities between words. In this work we use the word embeddings as the common vector space for both images and class labels and thus enable detection of objects from unseen categories.

Zero-shot image classification. Previous methods for tackling zero-shot classification used attributes, like shape, color, pose or geographical information as additional sources of information [54, 117, 118]. More recent approaches have used multimodal embeddings to learn a compatibility function between an image vector and class label embeddings [9, 10]. In [221], the authors augment the bilinear compatibility model by adding latent variables. The deep visual-semantic embedding model [55] used labeled image data and semantic information from unannotated text data to classify previously unseen image categories. We follow a similar methodology of using labeled object bounding boxes and semantic information in the form of unsupervised word embeddings to detect novel object categories. For a more compehensive overview of zero-shot classification, we refer the reader to the detailed

survey by Fu et al. [57].

Object detection. Early object detection approaches generated object proposals for each image and classified those object proposals using an image classification CNN [66, 67, 175]. More recent approaches use a single pass through a deep convolution network without the need for object region proposals [134, 173]. Recently, Redmon et al. [174] introduced an object detector which can scale upto 9000 object categories using both bounding box and image-level annotations. Unlike this setting, we work in a more challenging setting and do not observe any labels for the test object classes during training. We build our detection framework on an approach similar to the proposal-based approaches mentioned above.

Multi-modal learning. Using multiple modalities as additional sources of information has been shown to improve performance on several computer vision and machine learning tasks. These methods can be used for cross-modal retrieval tasks [48], or for transferring classifiers between modalities. Recently, [15] used images, text, and sound for generating deep discriminative representations which are shared across the three modalities. Similarly, [244] used images and text descriptions for improved natural language based visual entity localization. In [79], the authors used a shared vision and language representation space to obtain image-region and word descriptors that can be shared across multiple vision and language domains. Our work also uses multi-modal learning for building a robust object detector for unseen classes. Another related work is by Li et al. [125], which learns object-specific attributes to classify, segment, and predict novel objects. The problem proposed here differs considerably from this in detecting a large set of objects in unconstrained settings

and does not rely on using attributes.

Comparison with recent works on ZSD: Two concurrent works by Zhu et al. [251] and Rahman et al. [165] that address a similar problem. Zhu et al. focus on a different problem of generating object proposals for unseen objects. Rahman et al. [165] propose a loss formulation that combines max-margin learning and a semantic clustering loss. Their aim is to separate individual classes and reduce the noise in semantic vectors. A key difference between our work and Rahman et al. is the choice of evaluation datasets. Rahman et al. use the ILSVRC-2017 detection dataset [177] for training and evaluation. This dataset is more constrained in comparison to the ones used in our work (MSCOCO and VisualGenome) because it contains only about one object per image on an average. We would also like to note that due to a relatively simpler test setting, Rahman et al. does not consider the corrruption of the background class by unseen classes as done in this work and by Zhu et al.

3.2 Approach

We first outline our baseline ZSD approach that adapts prior work on zero-shot learning for the current task. Since this approach does not consider the diversity of the background objects during training, we then present an approach for training a background-aware detector with a fixed background class. We highlight some possible limitations of this approach and propose a latent assignment based background-aware model. Finally, we describe our method for densely sampling

labels using additional data, which improves generalization.

3.2.1 Baseline Zero-Shot Detection (ZSD)

We denote the set of all classes as $\mathcal{C} = \mathcal{S} \cup \mathcal{U} \cup \mathcal{O}$, where $\mathcal{S}$ denotes the set of seen (train) classes, $\mathcal{U}$ the set of unseen (test) classes, and $\mathcal{O}$ the set of classes that are neither part of seen or unseen classes. Note that our methods do not require a pre-defined test set. We fix the unseen classes here just for quantitative evaluation. We work in a zero-shot setting for object detection where, during training we are provided with labeled bounding boxes that belong to the seen classes only, while during testing we detect objects from unseen classes. We denote an image as $I \in \mathbb{R}^{M \times N \times 3}$, provided bounding boxes as $b_i \in \mathbb{N}^4$, and their associated labels as $y_i \in \mathcal{S}$. We extract deep features from a given bounding box obtained from an arbitrary region proposal method. We denote the extracted deep features for each box b_i as $\phi(b_i) \in \mathbb{R}^{D_1}$. We use semantic embeddings to capture the relationships between seen and unseen classes and thus transfer a model trained on the seen classes to the unseen classes as described later. We denote the semantic embeddings for different class labels as $w_j \in \mathbb{R}^{D_2}$, which can be obtained from pre-trained word embedding models such as Glove [158] or fastText [100]. Our approach is based on visual-semantic embeddings where both image and text features are embedded in the same metric space [55, 193]. We project features from the bounding box to the semantic

embedding space itself via a linear projection,

$$\psi_i = W_p \phi(b_i) \tag{3.1}$$

where, $W_p \in \mathbb{R}^{D_2 \times D_1}$ is a projection matrix and ψ_i is the projected feature. We use the common embedding space to compute a similarity measure between a projected bounding box feature ψ_i and a class embedding w_j for class label y_j as the cosine similarity S_{ij} between the two vectors. We train the projection by using a max-margin loss which enforces the constraint that the matching score of a bounding box with its true class should be higher than that with other classes. We define loss for a training sample b_i with class label y_i as,

$$\mathcal{L}(b_i, y_i, \theta) = \sum_{j \in \mathcal{S}, j \neq i} \max(0, m - S_{ii} + S_{ij}) \tag{3.2}$$

where θ refers to the parameters of the deep CNN and the projection matrix, and m is the margin. We also add an additional reconstruction loss to $\mathcal{L}$, as suggested by Kodirov et al. [108], to regularize the semantic embeddings. In particular, we use the projected box features to reconstruct the original deep features and calculate the reconstruction loss as the squared $L2$-distance between the reconstructed feature and the original deep feature. During test we predict the label ($\hat{y}_i$) for a bounding box (b_i) by finding its nearest class based on the similarity scores with different class

embeddings, i.e.

$$\hat{y}_i = \arg\max_{j \in \mathcal{U}} S_{ij} \tag{3.3}$$

It is common for object detection approaches to include a background class to learn a robust detector that can effectively discriminate between foreground and background objects. This helps in eliminating the bounding box proposals that clearly do not contain any objects of interest. We refer to these models as background-aware detectors. However, selecting a background for ZSD is a non-trivial problem as we do not know if a given background box includes background "stuff" in the classical sense e.g. sky, ground etc. or an instance of an unseen object class. We thus train our first (baseline) model only on bounding boxes that contain seen classes.

3.2.2 Background-Aware Zero-Shot Detection

While background boxes usually lead to improvements in detection performance for current object detection methods, for ZSD to decide which background bounding boxes to use is not straight-forward. We outline two approaches for extending the baseline ZSD model by incorporating information from background boxes during training.

Statically Assigned Background (SB) Based Zero-Shot Detection. Our first background-aware model follows as a natural extension of using a fixed background class in standard object detectors to our embedding framework. We accomplish this

by adding a fixed vector for the background class in our embedding space. Such 'statically-assigned' background modeling in ZSD, while providing a way to incorporate background information, has some limitations. First, we are working with the structure imposed by the semantic text embeddings that represent each class by a vector relative to other semantically related classes. In such a case it is difficult to learn a projection that can map all the diverse background appearances, which surely belong to semantically varied classes, to a single embedding vector representing one monolithic background class. Second, even if we are able to learn such a projection function, the model might not work well during testing. It can map any unseen class to the single vector corresponding to the background, as it has learned to map everything, which is not from seen classes, to the singleton background class.

Latent Assignment Based (LAB) Zero-Shot Detection. We solve the problems discussed above by spreading the background boxes over the embedding space by using an Expectation Maximization (EM)-like algorithm. We do so by assigning multiple (latent) classes to the background objects and thus covering a wider range of visual concepts. This is reminiscent of semi-supervised learning algorithms [185]; we have annotated objects for seen classes and unlabeled boxes for the rest of the image regions. At a higher level we encode the knowledge that a background box does not belong to the set of seen classes ($\mathcal{S}$), and could potentially belong to a number of different classes from a large vocabulary set, referred to as background set and denoted as $\mathcal{O}$.

We first train a baseline ZSD model on boxes that belong to the seen classes. We then follow an iterative EM-like training procedure (Algorithm 1), where, in the

Algorithm 1 LAB algorithm

Given: `annoData` (annotated data), `bgData` (background/unannotated data), $\mathcal{C}$ (set of all classes), $\mathcal{S}$ (seen classes), $\mathcal{U}$ (unseen classes), $\mathcal{O}$ (background set), `initModel` (pre-trained network)

```
currModel ← train(initModel, annoData)
for i = 1 to niters do
    currBgData ← φ
    for b in bgData do
        // distribute background boxes over open vocabulary minus seen classes
        b_new ←predict(b, currModel, O)
        // O = C \ (S ∪ U)
        currBgData ← currBgData ∪{b_new}
    end for
    currAnnoData ← annoData ∪ currBgData
    currModel←train(currModel,currAnnoData)
end for
return currModel
```

first of two alternating steps, we assign labels to some randomly sampled background boxes in the training set as classes in $\mathcal{O}$ using our trained model with Equation 3.3. In the second step, we re-train our detection model with the boxes, labeled as above, included. In the next iteration, we repeat the first step for another part of background boxes and retrain our model with the new training data. This proposed approach is also related to open-vocabulary learning where we are not restricted by a fixed set of classes [94, 241], and to latent-variable based classification models e.g. [186].

3.2.3 Densely Sampled Embedding Space (DSES)

The ZSD method, described above, relies on learning a common embedding space that aligns object features with label embeddings. A practical problem in learning such a model with small datasets is that there are only a small number of seen

classes, which results in a sparse sampling of the embedding space during training. This is problematic particularly for recognizing unseen classes which, by definition, lie in parts of the embedding space that do not have training examples. As a result the method may not converge towards the right alignment between visual and text modalities. To alleviate this issue, we propose to augment the training procedure with additional data from external sources that contain boxes belonging to classes other than unseen classes, $y_i \in \mathcal{C} - \mathcal{U}$. In other words, we aim to have a dense sampling of the space of object classes during training to improve the alignment of the embedding spaces. We show empirically that, because the extra data being used is from diverse external sources and is distinct from seen and unseen classes, it improves the baseline method.

3.3 Experiments

We first describe the challenging public datasets we use to validate the proposed approaches, and give the procedure for creating the novel training and test splits[1]. We then discuss the implementation details and the evaluation protocol. Subsequently, we present the empirical performance for different models followed by some ablation studies and qualitative results to provide insights into the methods.

MSCOCO [130] We use training images from the 2014 training set and randomly sample images for testing from the validation set.

VisualGenome (VG) [112] We remove non-visual classes from the dataset; use images from part-1 of the dataset for training, and randomly sample images from

[1]Visit `http://ankan.umiacs.io/zsd.html`

part-2 for testing.

OpenImages (OI) [111] We use this dataset for densely sampling the label space as described in Section 3.2.3. It contains about 1.5 million images containing 3.7 million bounding boxes that span 545 object categories.

Procedure for Creating Train and Test Splits: For dividing the classes into seen (train) and unseen (test) classes, we use a procedure similar to [13]. We begin with word-vector embeddings for all classes and cluster them into K clusters using cosine similarity between the word-vectors as the metric. We randomly select 80% classes from each cluster and assign these to the set of seen classes. We assign the remaining 20% classes from each cluster to the test set. We set the number of clusters to 10 and 20 for MSCOCO and VisualGenome respectively. Out of all the available classes, we consider only those which have a synset associated with them in the WordNet hierarchy [148] and also have a word vector available. This gives us 48 training classes and 17 test classes for MSCOCO and 478 training classes and 130 test classes for VisualGenome. For MSCOCO, to avoid taking unseen categories as background boxes, we remove all images from the training set which contain any object from unseen categories. However, we can not do this for VG because the large number of test categories and dense labeling results in most images being eliminated from the training set. After creating the splits we have $73,774$ training and $6,608$ test images for MSCOCO, and $54,913$ training and $7,788$ test images for VG.

3.3.1 Implementation Details

Preparing Datasets for Training: We first obtain bounding box proposals for each image in the training set. We construct the training datasets by assigning each proposal a class label from seen classes or the "background" class based on its IoU (Intersection over Union) with a ground truth bounding box. Since, majority of the proposals belong to background, we only include a part of the background boxes. Any proposal with $0 < \text{IoU} < 0.2$ with a ground truth bounding box is included as a background box in the training set. Apart from these, we also include a few randomly selected background boxes with $\text{IoU} = 0$ with any ground truth bounding boxes. Any proposal with an $\text{IoU} > 0.5$ with a ground-truth box is assigned to the class of the ground-truth box. Finally, we get 1.4 million training boxes for MSCOCO and 5.8 million training boxes for VG. We use these boxes for training the two background aware models. As previously mentioned, we only use boxes belonging to seen classes for training the baseline ZSD model. In this case, we have 0.67 million training boxes for MSCOCO and about 2.6 million training boxes for VG. We train our model on these training sets and test them on the test sets as described above.

Baseline ZSD Model: We build our ZSD model on the RCNN framework that first extracts region proposals, warps them, and then classifies them. We use the Edge-Boxes method [257] with its default parameters for generating region proposals and then warp them to an image of size 224×224. We use the (pre-trained) Inception-ResNet v2 model [198] as our base CNN for computing deep features. We project

image features from a proposal box to the 300 dimensional semantic text space by adding a fully-connected layer on the last layer of the CNN. We use the Adam optimizer [106] with a starting learning rate of 10^{-3} for the projection matrix and 10^{-5} for the lower layers. The complete network, including the projection layer, is first pre-trained on the MSCOCO dataset with the test classes removed for different models and datasets. For each algorithm, we perform end-to-end training while keeping the word embeddings fixed. The margin for ranking loss was set to 1 and the reconstruction loss was added to max-margin loss after multiplying it by a factor of 10^{-3}. We provide algorithm specific details below.

Static Background based ZSD: In this case, we include the background boxes obtained as described above in the training set. The single background class is assigned a fixed label vector $[1, \ldots, 0]$ (this fixed background vector was chosen so as to have norm one similar to the other class embeddings).

LAB: We first create a vocabulary ($\mathcal{C}$) which contains all the words for which we have word-vectors and synsets in the WordNet hierarchy [148]. We then remove any label from seen and unseen classes from this set. The size of the vocabulary was about $82K$ for VG and about $180K$ for MSCOCO. In the first iteration, we use our baseline ZSD model to obtain labels from the vocabulary set for some of the background boxes. We add these boxes with the newly assigned labels to the training set for the next iteration (see Algorithm 1). We fine-tune the model from the previous iteration using this new training set for about one epoch. During our experiments we iterate over this process five times. Our starting learning rates were the same as above and we decreased them by a factor of 10 after every 2 iterations.

Dense Sampling of the Semantic Space: To increase label density, we use additional data from OI to augment the training sets for both VG and MSCOCO. We remove all our test classes from OI and add the boxes from remaining classes to the training sets. This led to the addition of 238 classes to VG and 330 classes to MSCOCO during training. This increases the number of training bounding boxes for VG to 3.3 million and to 1 million for MSCOCO.

3.3.2 Evaluation Protocol

During evaluation we use Edge-Boxes for extracting proposals for each image and select only those proposals that have a proposal score (given by Edge-Boxes) greater than 0.07. This threshold was set based on trade-offs between performance and evaluation time. We pass these proposals through the base CNN and obtain a score for each test class as outlined in Section 3.2.1. We apply greedy non-maximal suppression [67] on all the scored boxes for each test class independently and reject boxes that have an IoU greater than 0.4 with a higher scoring box. We use recall as the main evaluation metric for detection instead of the commonly used mean average precision (mAP). This is because, for large-scale crowd-sourced datasets such as VG, it is often difficult to exhaustively label bounding box annotations for all instances of an object. Recall has also been used in prior work on detecting visual relationships [137] where it is infeasible to annotate all possible instances. The traditional mAP metric is sensitive to missing annotations and will count such detections as false positives. (However, for MSCOCO we report the mAP too since

Table 3.1: $|\mathcal{S}|$, $|\mathcal{U}|$, and $|\mathcal{O}|$ refer to the number of seen, unseen and the average number of active background classes considered during training respectively. BG-aware means background-aware representations. This table shows Recall@100 performance for the proposed zero-shot detection approaches (see Section 3.2) on the two datasets at different IoU overlap thresholds with the ground-truth boxes. The numbers in parentheses are mean average precision (mAP) values for MSCOCO. The number of test (unseen) classes for MSCOCO and VisualGenome are 17 and 130 respectively.

		MSCOCO						**Visual Genome**					
ZSD Method	**BG-aware**	**#classes**			**IoU**			**#classes**			**IoU**		
		$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{U}\rvert$	$\lvert\mathcal{O}\rvert$	0.4	0.5	0.6	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{U}\rvert$	$\lvert\mathcal{O}\rvert$	0.4	0.5	0.6
Baseline		48	17	0	34.36	22.14 (0.32)	11.31	478	130	0	8.19	5.19	2.63
SB	✓	48	17	1	34.46	24.39 (0.70)	12.55	478	130	1	6.06	4.09	2.43
DSES		378	17	0	**40.23**	**27.19** (0.54)	**13.63**	716	130	0	7.78	4.75	2.34
LAB	✓	48	17	343	31.86	20.52 (0.27)	9.98	478	130	1673	**8.43**	**5.40**	**2.74**

all object instances in MSCOCO have been annotated.) We define Recall@K as the recall when only the top K detections (based on prediction score) are selected from an image. A predicted bounding box is marked as true positive only if it has an IoU overlap greater than a certain threshold t with a ground truth bounding box and no other higher confidence predicted bounding box has been assigned to the same ground truth box. Otherwise it is marked as a false positive.

3.3.3 Quantitative Results

We present extensive results (Recall@100) for different algorithms on MSCOCO and VG datasets in Table 3.1 for three different IoU overlap thresholds. We also show the number of seen, unseen, and background classes for each case. During our discussion we report Recall@100 at a threshold of IoU ≥ 0.5 unless specified otherwise.

On the VG dataset the baseline model achieves 5.19% recall and the static

background (SB) model achieves a recall of 4.09%. This marked decline in performance is because all the background boxes are being mapped to a single vector. In VG some of these background boxes might actually belong to the seen (train) or unseen (test) categories. This leads to the SB model learning sub-optimal visual embeddings. However, for MSCOCO we observe that the SB model increases the recall to 24.39% from the 22.14% achieved by the baseline model. This is because we remove all images that contain any object from unseen classes from the training set for MSCOCO. This precludes the possibility of having any background boxes belonging to the test classes in the training set. As a result, the SB model is not corrupted by non-background objects and is thus more robust than the baseline.

When we densely sample the embedding space and augment the training classes with additional data, the recall for MSCOCO increases significantly from 22.14% (for baseline) to 27.19%. This shows that dense sampling is beneficial for predicting unseen classes that lie in sparsely sampled parts of the embedding space. With dense sampling, the number of train classes in MSCOCO are expanded by a factor of 7.8 to 378. In contrast, VG *a priori* has a large set of seen classes (478 versus 48 in MSCOCO), and the classes expand only by a factor of 1.5 (716) when using DSES. As a result dense sampling is not able to improve the embedding space obtained by the initial set of categories. In such scenarios it might be beneficial to use more sophisticated methods for sampling additional classes that are not represented well in the training set [62, 127, 161].

The latent assignment based (LAB) method outperforms the baseline, SB, and DSES on VG. It achieves a recall of 5.40% compared to 5.19%, 4.09% and 4.75%

achieved by baseline, SB, and DSES respectively. The consistent improvement across all IoUs compared to SB, confirms the benefits of spreading background objects over the embedding space. However, LAB gives a lower performance compared to the baseline for MSCOCO (20.52% by LAB versus 22.14% by baseline). This is not surprising since the iterations for LAB initialize with a larger set of seen classes for VG as compared to MSCOCO, resulting in an embedding that covers a wider spectrum of visual space. As a result, LAB is able to effectively spread the background boxes over a larger set of classes for VG leading to better detections. On the other hand, for MSCOCO a sparsely sampled embedding space restricts the coverage of visual concepts leading to the background boxes being mapped to a few visual categories. We also see this empirically in at the average number of background classes (set $\mathcal{O}$) assigned to the background boxes during iterations for LAB, which were 1673 for VG versus 343 for MSCOCO. In the remainder of this chapter we focus on LAB method for VG and SB for MSCOCO due to their appropriateness for the respective datasets.

We observe that the relative class-wise performance trends are similar to object detection methods, such as Faster RCNN[2] trained on fully supervised data. For example, classes such as "bus" and "elephant" are amongst the best performing while "scissors" and "umbrella" rank amongst the worst in performance. In addition to these general trends, we also discover some interesting findings due to the zero-shot nature of the problem. For example, the class "cat", which generally performs well with standard object detectors, did not perform well with SB. This results from

[2] http://cocodataset.org/#detections-leaderboard

having an insufficient number of semantically related categories for this class in the training set which does not allow the model to effectively capture the appearance of class "cat" during testing. For such cases we find dense sampling to be useful during training. The class "cat" is one of the top performing categories with DSES. Based on such cases we infer that for ZSD the performance is both a function of appearance characteristics of the class as well as its relationship to the seen classes. For VG, the best performing classes, such as "laptop", "car", "building", "chair", seem to have well defined appearance characteristics compared to poorly performing classes, such as "gravel", "vent", "garden", which seem to be more of "stuff" than "things". We also observe that the model is unable to capture any true positive for the class "zebra" and is instead detecting instances of "zebra" as either "cattle" or "horse". This is because the model associates a "zebra" with a "giraffe", which is close in the semantic space. The model is able to adapt the detector for the class "giraffe" to the class "zebra" but fails to infer additional knowledge needed for a successful detector that a zebra differs from a giraffe in having white stripes, lower height, and has a body structure similar to a horse. Finally, we also observe that compared to the baseline, LAB achieves similar or better performance on 104 of 130 classes on VG. While for MSCOCO, SB and DSES achieve better or similar performance on 12 and 13 classes respectively out of 17 classes, highlighting the advantages of the proposed models.

3.3.4 Generalized Zero-Shot Detection (GZSD)

The generalized zero-shot learning setting is more realistic than the previously discussed zero-shot setting [222] because both seen and unseen classes are present during evaluation. This is more challenging than ZSD because it removes the prior knowledge that the objects at test time belong to unseen classes only. We use a simple novelty detection step which does not need extra supervision. Given a test bounding box, b_i, we first find the most probable train and test classes (see (3.3)) ($\hat{y}_i^s$ and $\hat{y}_i^u$ respectively) and the corresponding similarity scores (s_i and u_i). As the novelty detection step, we check if u_i is greater than some threshold n_t. We assign the given bounding box to class $\hat{y}_i^u$ if $u_i \geq n_t$, otherwise to $\hat{y}_i^s$. For MSCOCO, DSES gives the best performance in the GZSD setting too. At $n_t = 0.2$, DSES achieves a Recall@100 of 15.02% for seen classes and 15.32% for unseen classes (harmonic mean 15.17% [222]) at $IoU \geq 0.5$ compared to 14.54% and 10.57% (HM 12.24%) for the LAB model and 16.93% and 8.91% (HM 11.67%) for baseline.

3.3.5 Ablation Studies

We compare results when considering different number, K, of high-confidence detections.We define $K = All$ as the scenario where we consider all boxes returned by the detector with a confidence score greater than the threshold for evaluation. We compare LAB and the SB models for VG and MSCOCO respectively, with the corresponding baseline models in Table 3.2.

The difference in performance between the cases $K = All$ and $K = 100$ is

Table 3.2: Ablation studies on background-aware approaches for ZSD. We highlight results where the performance is higher for background-aware approaches compared to the corresponding baseline. For MSCOCO, the values in parentheses are mAP.

	MSCOCO						VisualGenome					
	Baseline			SB			Baseline			LAB		
K↓ IoU→	0.3	0.4	0.5	0.3	0.4	0.5	0.3	0.4	0.5	0.3	0.4	0.5
All	47.91	37.86	24.47 (0.22)	43.79	35.58	**25.12** (0.64)	13.88	9.98	6.45	12.75	9.61	6.22
100	43.62	34.36	22.14 (0.32)	42.22	**34.46**	**24.39** (0.70)	11.34	8.19	5.19	11.20	**8.43**	**5.40**
80	41.69	32.64	21.01 (0.38)	41.47	**33.98**	**24.01** (0.72)	10.41	7.55	4.75	**10.45**	**7.86**	**5.06**
50	36.19	27.37	17.05 (0.50)	**39.82**	**32.6**	**23.16** (0.81)	7.98	5.79	3.68	**8.54**	**6.44**	**4.14**

small, in general, for the background-aware algorithms unlike the baseline. For example, on MSCOCO the recall for SB falls by an average (across IoUs) of 1.14% points, compared to a fall of 3.37% for the baseline. This trend continues further down to $K = 80$ and $K = 50$ with a gradual decline in performance as K decreases. This shows that high confidence detections from our model are of high quality.

We observe that the background-aware models give better quality detections compared to baselines. The Recall@K for the corresponding background-aware models are better than the baseline at lower K and higher IoU threshold values for both datasets. This region represents higher quality detections.This shows that incorporating knowledge from background regions is an important factor for improving detection quality and performance for ZSD.

3.3.6 Qualitative Results

Figure 3.2 shows output detections by the background aware models, i.e. LAB on VisualGenome (first two rows) and SB on MSCOCO (last row). Blue boxes show correct detections and red boxes show false positives. These examples confirm that

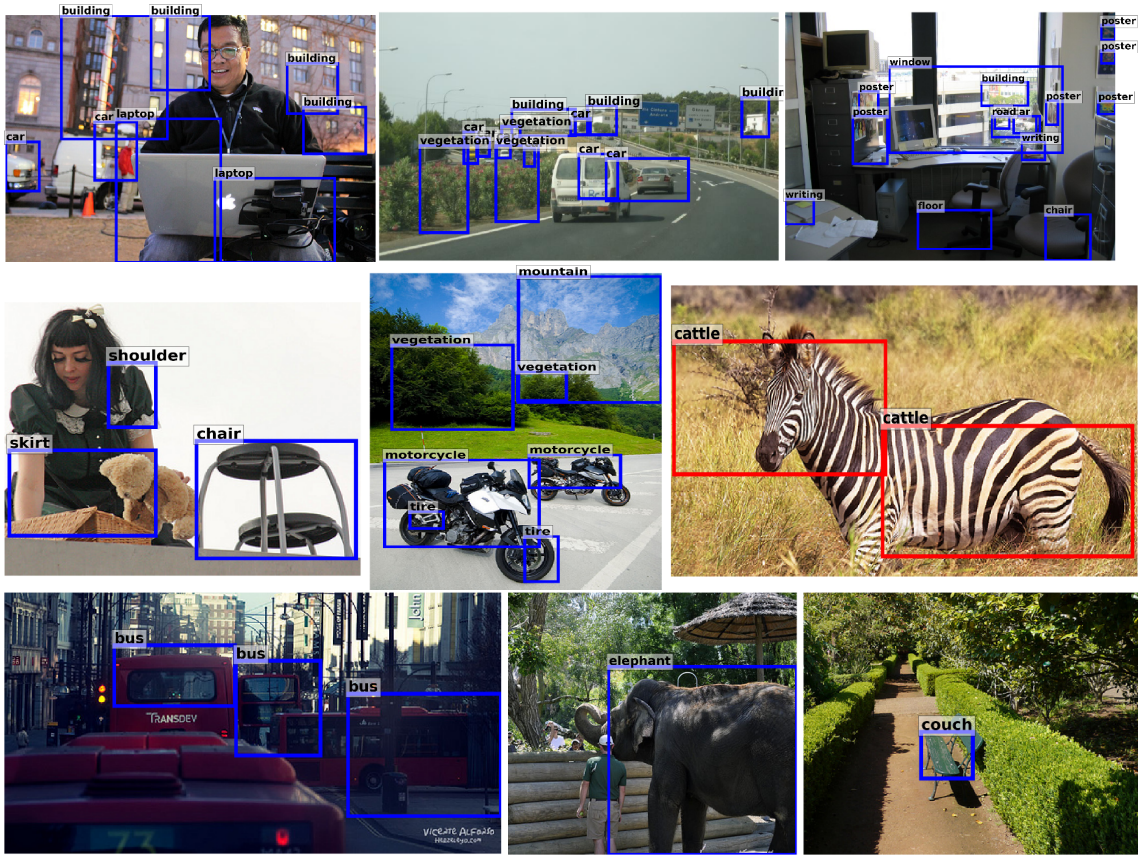


Figure 3.2: This figure shows some detections made by the background-aware methods. We have used Latent Assignment Based model for VisualGenome (rows $1-2$) and the Static Background model (row 3) for MSCOCO. Reasonable detections are shown in blue and two failure cases in red.

the proposed models are able to detect unseen classes without observing any samples during training. Further, the models are able to successfully detect multiple objects in real-world images with background clutter. For example, in the image taken in an office (1^{st} row 3^{rd} column), the model is able to detect object classes such as "writing", "chair", "cars". It is also interesting to note that our approach understands and detects "stuff" classes such as "vegetation", and "floor". As discussed in Section 3.3.3, we have shown a failure case "zebra", that results from having limited information regarding the fine-grained differences between seen and unseen classes.

3.4 Discussion and Conclusion

We used visual-semantic embeddings for ZSD and addressed the problems associated with the framework which are specific to this problem. We proposed two background-aware approaches; the first one uses a fixed background class while the second iteratively assigns background boxes to classes in a latent variable framework. We also proposed to improve the sampling density of the semantic label space using auxiliary data. We proposed novel splits of two challenging public datasets, MSCOCO and VisualGenome, and gave extensive quantitative and qualitative results to validate the methods proposed.

Chapter 4: Detecting Human-Object Interactions using Functional Generalization

Human-object interaction detection is the task of localizing and inferring relationships between a human and an object, e.g., "eating an apple" or "riding a bike." Given an input image, the standard representation for HOIs [30, 69, 76, 178] is a triplet ⟨`human`, `predicate`, `object`⟩, where `human` and `object` are represented by bounding-boxes, and `predicate` is the interaction between this (`human`, `object`) pair. At first glance, it seems that this problem is a composition of the atomic problems of object detection [66, 134, 173, 175] (independently localizing humans and objects) and classification [69, 187] (post-hoc classifying their interaction). These atomic recognition tasks are certainly the building blocks of a variety of approaches for HOI understanding [40, 69, 187]; and the progress in these atomic tasks directly translates to improvements in HOI understanding. However, the task of HOI understanding comes with its own unique set of challenges [30, 137, 178].

These challenges are due to the combinatorial explosion of the possible interactions with increasing number of objects and predicates. For example, in the commonly used HICO-Det dataset [30] with 80 unique object classes and 117 predicates, there are 9,360 possible relationships. This number increases to more than

Figure 4.1: Illustration of common properties of HOI Detection. (Top row) Datasets are not exhaustively labeled. (Bottom row) Humans interact in a similar fashion with functionally similar objects - both persons could be eating either a burger, a hot dog, a sandwich, or a pizza.

10^6 for larger datasets like Visual Genome [112] and HCVRD [256], which have hundreds of object categories and thousands of predicates. This, combined with the long-tail distribution of HOI categories, makes it difficult to collect labeled training data for all HOI triplets. A common solution to this problem is to arbitrarily limit the set of HOI relationships and only collect labeled images for this limited subset. For example, the HICO-Det benchmark has only about 600 unique relationships.

Even though these datasets can be used for training fully-supervised models for recognizing a limited set of HOI triplets, they do not address the problem completely. For example, consider the images shown in Figure 4.1 (top row) from the challenging HICO-Det dataset. The three pseudo-synonymous relationships: ⟨`human`, `hold`,

bicycle⟩, ⟨human, sit_on, bicycle⟩, and ⟨human, straddle, bicycle⟩ are all possible for both these images; but only a subset is labeled for each. We argue that this is not a quality control issue while collecting a dataset, but a problem associated with the huge space of possible HOI relationships. It is enormously challenging to exhaustively label even the 600 unique HOIs, let alone all the possible interactions among humans and objects. Any HOI detection model that relies entirely on labeled data will be unable to recognize the relationship triplets that are not present in the dataset, but are common in the real-world. For example, a naïve model trained on HICO-Det cannot recognize ⟨human, push, car⟩ triplet because this triplet does not exist in the training set. The ability to recognize previously unseen relationships (zero-shot recognition) is a highly desirable capability for a HOI detection system.

In this work, we address the challenges discussed above using a model that leverages the common-sense knowledge that humans have similar interactions with objects that are functionally similar. The proposed model has an inherent ability to do zero-shot detection. Consider the images in Figure 4.1 (second row) with ⟨human, eat, ?⟩ triplet. The person in either image could be eating a burger, a sandwich, a hot dog, or a pizza. Inspired by this, our key contribution is incorporating this common-sense knowledge in a model for generalizing HOI detection to functionally similar objects. This model utilizes visual appearance of a human, their relative geometry with the object, and language priors [147] to determine which objects afford similar predicates [64]. Such a model is able to exploit the large amount of contextual information present in language priors to generalize HOIs across functionally similar objects.

In order to train this module, we need a list of functionally similar objects and labeled examples for the relevant HOI triplets, neither of which are readily available. To overcome this, we propose a way to train this model by: 1) using a large vocabulary of objects, 2) discovering functionally similar objects automatically, and 3) proposing data-augmentation, emulating the examples shown in Figure 4.1 (second row). To discover functionally similar objects in an unsupervised way, we use a combination of visual appearance features [85] and semantic word embeddings [147] to represent the objects in a "world set" (Open Images Dataset (OID) [111] in this work). Note that the proposed method is not contingent on the world set. Any large dataset, like ImageNet [177], could replace the open images dataset. Finally, to emulate the examples shown in Figure 4.1 (second row), we use the human and object bounding-boxes from a labeled interaction, the visual features from the human bounding-box, and semantic word embeddings of all functionally similar objects. Notice that this step does not utilize the visual features for objects, just their relative locations with respect to a human, enabling us to perform this data-augmentation.

The proposed approach achieves over 7% absolute improvement in mAP over the best published method for HICO-Det. Further, using a generic object detector, the proposed functional generalization model lends itself directly to the zero-shot HOI triplet detection problem. We clarify that zero-shot detection is the problem of detecting HOI triplets for which the model has never seen any images. Knowledge about functionally similar objects enables our system to detect interactions involving objects not contained in the original training set. Using just this generic object

detector, our model achieves state-of-the-art performance for HOI detection on the popular HICO-Det dataset in the zero-shot setting, improving over existing methods by several percentage points. Additionally, we show that the proposed approach can be used as a way to deal with social/systematic biases present in image captioning and other vision+language datasets [12, 247].

In summary, our contributions are: (1) a functional generalization model for capturing functional similarities among objects; (2) a method for training the proposed model; and (3) state-of-the-art results on HICO-Det in both fully-supervised and zero-shot settings.

4.1 Related Work

Human-Object Interaction. Early methods [234–236] relied on structured visual features which capture contextual relationships between humans and objects. Similarly, [40] used structured representations and spatial co-occurrences of body parts and objects to train models for HOI recognition. Gupta et al. [74, 75] adopted a Bayesian approach that integrated object classification and localization, action understanding, and perception of object reaction. Desai et al. [43] constructed a compositional model which combined skeleton models, poselets, and visual phrases.

More recently, with the release of large datasets [30, 31, 76, 112, 256], the problem of detecting and recognizing HOIs has attracted signification attention. This has been driven by HICO [31] which is a benchmark dataset for recognizing human-object interactions. The HICO-Det dataset [30] extended HICO by adding bounding

box annotations. V-COCO [76] is a much smaller dataset containing 26 classes and about 10,000 images. On the other hand, HCVRD [256] and Visual Genome [112] provide annotations for thousands of relationship categories and hundreds of objects. However, they suffer from noisy labels. Therefore, we use the HICO-Det dataset to evaluate our approach.

Gkioxari et al. [69] designed a system which trains object and relationship detectors simultaneously on the same dataset and classifies a human-object pair into a fixed set of pre-defined relationship classes. This precludes the method from being useful for detecting novel relationships. Similarly, [227] used pose and gaze information for HOI detection. Kolesnikov et al. [109] introduced the Box Attention module to a standard R-CNN and trained simultaneously for object detection and relationship triplet prediction. Graph Parsing Neural Networks [162] incorporated structural knowledge and inferred a parse graph in a message passing inference framework. In contrast, our method does not need iterative processing and requires only a single pass through a small neural network.

Unlike most prior works, we do not directly classify into a fixed set of relationship triplets but into predicates. This helps us detect completely unseen interactions. The method which is the closest in spirit to our approach is [187]. The authors used a two branch structure where the first branch is responsible for detecting humans and predicting the predicates, and the second branch detects objects. Unlike the proposed approach, their method does not even consider the object while predicting the predicate. It solely depends on the appearance of the human. Also, they do not use any prior information from language. Our model utilizes implicit human pose,

the object label, human-object geometric relationship, and knowledge about similarities among the objects. Hence, we achieve much better performance than [187] using the combination of these factors.

We also distinguish our work from prior works [50, 65, 103, 139] on HOI recognition where the task is to recognize the interaction in an image but not to locate the actors and objects. We tackle the more difficult problem of detecting HOIs. However, similar to some work on HOI recognition, we also work with the idea of using language encoded by word vectors to train our generalization module.

Zero-shot Learning. Our work also ties well with recent work on zero-shot classification [29, 108, 118, 222] and the nascent field of zero-shot object detection [21, 41, 165]. In Chapter 3 we discussed an approach that projects images into the word-vector space to exploit the semantic properties of such spaces. A similar idea was used in [108] for zero-shot classification. Rahman et al. [165], on the other hand, used meta-classes to cluster semantically similar classes while keeping distinct classes separate. In this work, we also use word-vectors as additional semantic information to our generalization module. This, along with our approach for incorporating generalization during training, helps the model detect previously unseen HOIs.

4.2 Approach

Figure 4.2 represents our approach. The main novelty and contribution of the proposed approach lies in incorporating generalization through a language component. This is done by using functionally similar objects to train the model. During infer-

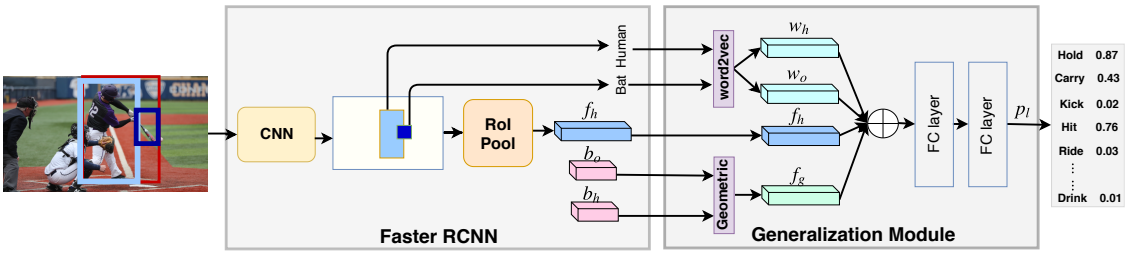


Figure 4.2: We detect all objects and humans in an image. This detector gives human features f_h, and the corresponding labels. We consider all pairs of human-object and create union boxes. Our functional generalization module uses the word vectors for the human w_h, the object class w_o, geometric features f_g, and f_h to produce the probability estimate over the predicates.

ence, we first detect humans and objects in the image using our object detectors, which also give the corresponding (RoI-pooled [175]) feature representations. Each detected human-object pair is used to extract visual and language features which are used to estimate the predicate associated with the interaction. We describe each component of the model in detail and the training procedure in the following sections.

4.2.1 Object Detection

For our experiment in the fully-supervised setting, we use an object detector fine-tuned on the HICO-Det dataset. For zero-shot HOI detection and additional experiments, we use a Faster-RCNN [92] -based detector trained on the Open Images dataset (OID) [111]. This network can detect 545 object categories and we use it to obtain proposals for humans and objects in an image. The object detectors also output the ROI-pooled [175] features corresponding to these detections. All human-object pairs thus obtained are passed to our model which outputs probabilities for each predicate.

4.2.2 Functional Generalization Module

Humans interact with objects that are functionally similar in similar ways. Leveraging this fact, the functional generalization module exploits object similarity encoded in word vectors, the relative spatial location of human and object boxes, and the implicit human appearance to estimate the predicate. At its core, it comprises an MLP, which takes as input the human and object word embeddings, w_h and w_o, the geometric relationship between the human and object boxes f_g, and the human visual feature f_h. The human embedding, w_h, helps in distinguishing between different words for humans (man/woman/boy/girl/person). The geometric feature is useful as the relative positions of a human and an object can help eliminate certain predicates. The human feature f_h is used as a representation for the appearance of the human. This appearance representation is added because the aim is to incorporate the idea that humans look similar while interacting with similar objects. For example, a person drinking from a cup looks similar while drinking from a glass or a bottle. The four features w_h, w_o, f_g, and f_h are concatenated and passed through the 2-layer MLP which predicts the probabilities for each predicate. All the predicates are considered independent. We now give details of different components in this model.

4.2.2.1 Word embeddings

We use 300-D vectors from word2vec [147] to get the human and object embeddings w_h and w_o. These encode semantic knowledge and allow the model to discover pre-

viously unseen interactions between a human and objects by exploiting the semantic similarities between objects.

4.2.2.2 Geometric features

Following prior work on visual relationship detection [255], we define the geometric relationship feature as:

$$f_g = \Bigg[\frac{x_1^h}{W}, \frac{y_1^h}{H}, \frac{x_2^h}{W}, \frac{y_2^h}{H}, \frac{A^h}{A^I}, \frac{x_1^o}{W}, \frac{y_1^o}{H}, \frac{x_2^o}{W}, \frac{y_2^o}{H}, \frac{A^o}{A^I}, \left(\frac{x_1^h - x_1^o}{x_2^o - x_1^o}\right), \left(\frac{y_1^h - y_1^o}{y_2^o - y_1^o}\right), \log\left(\frac{x_2^h - x_1^h}{x_2^o - x_1^o}\right), \log\left(\frac{y_2^h - y_1^h}{y_2^o - y_1^o}\right)\Bigg] \quad (4.1)$$

where, W, H are the image width and height, (x_i^h, y_i^h), and (x_i^o, y_i^o) are the human and object bounding box coordinates respectively, A^h is the area of the human box, A^o is the area of the object box, and A^I is the area of the image. The geometric feature f_g uses spatial features for both entities (human and object) and also spatial features from their relationship. It is a measure of the scales and relative positioning of the two entities.

4.2.2.3 Generalizing to new HOIs

We incorporate the idea that humans interacting with similar objects look similar through the functional generalization module. As shown in Figure 4.3, this idea can be added by changing the object name while keeping the human word vector w_h, the human visual feature f_h, and the geometric feature f_g fixed. Each object has

a different word-vector and the model learns to recognize the same predicate for different human-object pairs. Note that this does not need visual examples for all human-object pairs.

Finding similar objects. An obvious choice for defining similarity between objects would be to find the closest objects in the WordNet hierarchy [148]. However, this creates several issues which make using WordNet impractical/ineffective. The first is defining the distance between the nodes in the tree. The height of a node cannot be used as a metric because different things have different levels of categorization in the tree. Similarly, defining sibling relationships which adhere to functional intuitions is extremely challenging. Another major issue with using WordNet is the lack of correspondence between closeness in the tree and visual similarity between objects.

To overcome these problems, we consider similarity in both visual and semantic representations of objects. We start by defining a vocabulary of objects $\mathcal{V} = \{o_1, \ldots, o_n\}$ which includes all the objects that can be detected by our object detector. For each object $o_i \in \mathcal{V}$, we obtain a visual feature $f_{o_i} \in \mathbb{R}^p$ from images in OID, and a word vector $w_{o_i} \in \mathbb{R}^q$. We concatenate these two to obtain a mixed representation u_{o_i} for object o_i. We then cluster u_i's into K clusters using Euclidean distance. The objects in the same cluster are considered functionally similar. This clustering step has to be done only once. We use these clusters to find all objects similar to an object in the target dataset. Note that there might not be any visual examples for many of the objects obtained using this method. This is why we do not use the RoI-pooled visual features from the object.

We would like to point out that using either just the word2vec representations

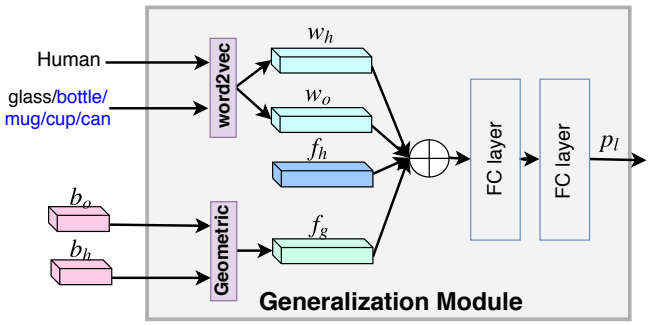


Figure 4.3: Generalization module. We can replace "glass" by "bottle", "mug", "cup", or "can".

or just the visual representations for clustering gave several inconsistent clusters. Therefore, we use the concatenated features u_{o_i}. We observed that the clusters created using these features better correspond to functional similarities between objects.

Generating training data. For each relationship triplet $<h,p,o>$ in the original dataset, we add r triplets $<h,p,o_1>$, $<h,p,o_2>$, ..., $<h,p,o_r>$ to the dataset keeping the human, and object boxes fixed, and only changing the object name. This means that, for all these f_g and f_h are the same as for the original sample. The r different objects, o_1,..., o_r belong to the same cluster as object o. For example, in Figure 4.3, the ground truth category "glass" can be replaced by "bottle", "mug", "cup", or "can" while keeping w_h, f_h, and f_g fixed.

4.2.3 Training

A training batch consists of T interaction triplets. The model produces the probabilities for each predicate independently. We use a weighted class-wise binary cross entropy loss for training the model.

We now describe a weighing strategy that can reduce the effects of lack of available exhaustive labels.

Noisy labeling. Missing and incorrect labels are a common issue in HOI datasets. Also, a human-object pair can have several different interactions at the same time. For example, a person can be sitting on a bicycle, riding a bicycle, and straddling a bicycle. These interactions are usually labeled with slightly different bounding boxes. To overcome these issues, we use a per-triplet loss weighing strategy. A training triplet in our dataset has a single label, e.g. <`human, ride, bicycle`>. A triplet with slightly shifted bounding boxes might have another label, like <`human, sit_on, bicyle`>. The idea is that the models should be penalized more if they fail to predict the correct class for a triplet. Given the training sample <`human, ride, bicycle`>, we want the model to definitely predict "`ride`", but we should not penalize it if it predicts "`sit_on`" as well. Therefore, while training the model, we use the following weighing scheme for classes. Suppose that a training triplet is labeled <`human, ride, bicycle`> and there are other triplets in the image. For the training triplet under consideration, we assign a high weight to the loss for the correct class (`ride`), and a zero weight to all other predicates in the image. We also scale down the weight to the loss for all other classes to ensure that the model is not penalized too much for predicting a missing but correct label.

The final step of inference is class-wise non-maximal suppression (NMS) of the union bounding boxes (union of human and object boxes). This helps in removing multiple detections for the same interaction and leads to higher precision at the same recall.

4.3 Experiments

We evaluate our approach on the large-scale HICO-Det dataset [30].

4.3.1 Dataset and Evaluation Metrics

HICO-Det extends the HICO (Humans Interacting with Common Objects) dataset [31] which contains 600 HOI categories for 80 objects. HICO-Det gives bounding box annotations for humans, and objects for each HOI category. The training set of HICO-Det contains over 38,000 images and about 120,000 HOI annotations for 600 HOI classes. The test set has 9,600 images and 33,400 HOI instances.

For evaluation, HICO-Det uses the mean average precision (mAP) metric commonly used in object detection [47, 130]. Here, a HOI detection is counted as a true positive if the minimum of the human overlap IOU_h and object overlap IOU_o with the ground truth is greater than 0.5. Performance is usually reported for three different HOI category sets: (a) all 600 HOI categories (Full), (b) 138 categories with less than 10 training samples (Rare), and (c) 462 categories with more than 10 training samples (Non-Rare).

4.3.2 Implementation Details

We start with a Faster-RCNN-based object detector which is fine-tuned for the HICO-Det dataset. The base network for this detector is a ResNet-101. This detector was originally trained on the COCO dataset [130] which has the same 80 object categories as the HICO-Det dataset. We consider all detections for which

the detection confidence is greater than 0.9 and create human-object pairs for each image. Each detection has an associated feature vector. These pairs are then passed through our model. The human feature f_h is 2048 dimensional. The two hidden layers in the model are of dimensions 1024 and 512. The model outputs probability estimates for each predicate and the final output prediction is all predicates with probability ≥ 0.5.

For all the experiments, we train the complete model for 25 epochs with an initial learning rate of 0.1 which is dropped by one-tenth every 10 epochs. We re-iterate that the object detector and the word2vec vectors are frozen while training this model. For all experiments we use upto five (r) additional objects for data augmentation. That is, for each human-object pair in the training set, we add upto five objects from the same cluster while leaving the bounding boxes and human features unchanged. We also describe zero-shot experiments where we show that our method can be used to detect previously unseen interactions.

4.3.3 Results

The last row in Table 4.1 show the results obtained by our model. We observe that our model comprehensively outperforms all existing methods. It achieves an mAP of 21.96%, an almost 7% absolute improvement over the best published method [59] and even over 2.5% over the best contemporary work [159]. Also note the performance for rare classes. Our model achieves 16.43% mAP for rare classes compared to the existing best of 15.40%. The performance, along with the simplicity,

Table 4.1: mAPs (%) in the default setting for the HICO-Det dataset. Our model was trained with upto five neighbors for each object. The last column is the total number of parameters in the models.

Method	**Full** (600 classes)	**Rare** (138 classes)	**Non-rare** (462 classes)	**# Params** (millions)
Shen et al. [187]	6.46	4.24	7.12	-
HO-RCNN + IP [30]	7.30	4.68	8.08	-
HO-RCNN + IP + S [30]	7.81	5.37	8.54	-
InteractNet [69]	9.94	7.16	10.77	-
iHOI [227]	9.97	7.11	10.83	-
GPNN [162]	13.11	9.34	14.23	-
ICAN [59]	14.84	10.45	16.15	48.1 + 40.9 = 89.0
Gupta et al. [78]	17.18	12.17	18.68	9.2 + 63.7 = 72.9
Interactiveness Prior [124]	17.22	13.51	18.32	35.0 + 29.0 = 64.0
Peyre et al. [160]	19.40	15.40	20.75	21.8 + 40.9 = 62.7
Ours	**21.96**	**16.43**	**23.62**	3.1 + 48.0 = **51.1**

of our model is a remarkable strength and reveals that existing methods may be over-engineered.

Comparison of number of parameters

In Table 4.1, we also compare the number of parameters in the four closest existing models against our model. With far fewer parameters, our model achieves better performance. For example, compared to the current state-of-the-art model which contains 65.1 million parameters and achieves only 19.40% mAP, our model contains just 51.1 million parameters and reaches an mAP of 21.96%. Ignoring the object detectors, our model introduces just 3.1 million new parameters. Note that [78] and [124] also include a pose estimation model. The number of parameters in Table 4.1 do not include pose estimation models. Our method provides a simple and intuitive way of thinking about the problem.

Next, we show how a generic object detector can be used to detect novel interactions, even those involving objects not present in the training set. For this, we use an off-the-shelf Faster RCNN-based object detector which is trained on the OpenImages dataset and is capable of detecting 545 object categories. This detector uses an Inception ResNet-v2 with atrous convolutions as its base network.

4.3.4 Zero-shot HOI Detection

Recent work on zero-shot learning aims to either recognize [222] or detect [21] previously unseen objects in images. Shen et al. [187] take this idea further and try to detect previously unseen human-object relationships in images. This means that the aim is to detect interactions for which no images are available during training. In this section, we show that our method offers significant improvements over [187] for zero-shot HOI detection.

4.3.4.1 Seen object scenario

We first consider the same setting as [187]. We select 120 relationship triplets ensuring that every object involved on these 120 relationships occurs in at least one of the remaining 480 triplets. We call this the "seen object" zero-shot setting, i.e., the model sees all the objects involved but not the exact relationships. Later, we will consider the "unseen object" setting as well where no relationships involving a particular set of objects will be observed during training.

Table 4.2 shows the performance of our approach in the "seen object" setting

Table 4.2: mAPs (%) in the default setting for ZSD. This is the seen object setting, i.e., it assumes that all the objects have been seen.

Method	**Unseen** (120 classes)	**Seen** (480 classes)	**All** (600 classes)
Shen et al. [187]	5.62	-	6.26
Ours	**10.93**	12.60	**12.26**

for 120 unseen triplets during training. We achieve significant improvement (5.3% absolute mAP) over the prior method for zero-shot interaction detection. Overall, on all 600 classes, our model gives 6% absolute improvement in mAP.

4.3.4.2 Unseen object scenario

Now we introduce the "unseen object" setting for evaluating zero-shot HOI detection. We start by randomly selecting 12 objects from the 80 objects in HICO. We pick all the relationships containing these objects. This gives us 100 relationship triplets which constitute the test (unseen) set for zero-shot HOI detection. We train models using visual examples from only the remaining 500 categories. Table 4.3 gives results for our methods in this setting. We cannot compare with existing methods because none of them have the ability to detect HOIs in the unseen object scenario. We hope that our method will serve as a baseline for future research on this important problem.

In Figure 4.4, we show that our model can detect interactions with objects for which no images are seen during training. This is because we use a generic detector which can detect many more objects. We note, here, that there are some

Table 4.3: mAPs (%) in the unseen object setting for ZSD. This is the unseen object setting where the trained model for interaction recognition has not seen any examples of some object classes.

Method	**Unseen** (100 classes)	**Seen** (500 classes)	**All** (600 classes)
Ours	11.22	14.36	13.84

Figure 4.4: Some HOI detections in the unseen object ZSD setting. Our model has not seen any image with the objects shown above during training. The first two rows are correct detections. While the last row shows some mistakes. Many of the incorrect detections are just slightly off from being correct. For example in the first image in the last row, the person is actually eating a pizza slice. However, because our model does not have the ability to reason in 3D, it cannot distinguish between a pizza in the foreground vs the pizza in the background.

classes among the 80 COCO classes which do not occur in OI. We willingly take the penalty for missing interactions involving these objects in order to present a more robust system which not only works for the dataset of interest but is able to generalize to completely unseen interaction classes (even the object was not seen). We believe that these are strong baselines for this setting and we will release class lists and training sets to standardize evaluation for future methods. We reiterate

Figure 4.5: Some HOI detections in the unseen object ZSD setting but for objects outside the 80 object categories in HICO-Det.

that none of the previous methods has the ability to detect HOIs in this scenario. In Figure 4.5, we further show some examples of detections made by our models for some objects outside the 80 object categories in HICO-Det or COCO.

4.3.5 Ablation Analysis

We point out that the generic object detector used for zero-shot HOI detection can also be used in the supervised setting. For example, using this detector, we obtain an mAP of 14.35% on the Full set of the HICO-Det dataset. This is a competitive performance and is better than most published works (Table 4.1). This shows the strength of our generalization approach. In this section, we provide further analysis of our model with the generic object detector.

Number of neighbours. To demonstrate the effectiveness of generalization through our method, we vary the number of neighboring objects which are added to the dataset for each training instance. Table 4.4 shows the effect of using different num-

Table 4.4: HICO-Det performance (mAP %) of the model with different number of neighbors considered for the generalization module.

r (Number of objects)	**Full** (600 classes)	**Rare** (138 classes)	**Non-rare** (462 classes)
0	12.72	7.57	14.26
3	13.70	7.98	15.41
5	**14.35**	**9.84**	**15.69**
7	13.51	7.07	15.44

ber of neighbors. The baseline (first row) is when no additional objects are added. This is the case when we do not use any additional data and rely only on the interactions present in the original dataset. We successively add more neighboring objects to the training data and observe that the performance improves significantly. However, after about five additional neighbors, the performance starts to saturate because noise from clustering starts to make an impact. Because the clusters are not perfect, adding more neighbors can start becoming harmful. Also, the training times increase rapidly. Therefore, as a trade-off between training speed and test performance, we add five neighbors for each HOI instance in all our experiments.

Clustering method. To check if another clustering algorithm might be better, we create clusters using different algorithms. From Table 4.5 we observe that K-means clustering leads to the best performance. Hierarchical agglomerative clustering also gives close albeit lower performance.

Importance of features. Further ablation studies (Table 4.6) showed that removing f_g, or f_h from the functional generalization module leads to a reduction in performance. For example, training the model without the geometric feature f_g

Table 4.5: mAPs (%) for different clustering methods.

Clustering Algorithm	**Full** (600 classes)	**Rare** (138 classes)	**Non-rare** (462 classes)
K means	**14.35**	**9.84**	15.69
Agglomerative	14.05	7.59	**15.98**
Affinity Propagation	13.49	7.53	15.28

Table 4.6: Ablation studies (mAP %).

Setting	**Full** (600 classes)	**Rare** (138 classes)	**Non-rare** (462 classes)
Base	**14.35**	**9.84**	**15.69**
Base $-f_h$	12.15	4.87	14.33
Base $-f_g$	12.43	8.02	13.75

gives an mAP of 12.43% and training the model without f_h in the generalization module gives an mAP of just 12.15% showing the importance of both features in the model. In particular, note that the performance for rare classes in the absence of f_h is very low (4.87%). This shows that using visual information from the human is essential for detecting rare HOIs.

4.3.6 Dealing with Dataset Bias

Dataset bias leads to models being biased towards particular classes [202]. In fact, bias in the training dataset is usually amplified by the models [12,247]. Our proposed method can be used as a way to overcome the dataset bias problem. To illustrate this, we use metrics proposed in [247] to quantitatively study model bias.

Adopting the bias metric from [247], we define the bias for a object-verb pair,

(o, v_*) in a set as:

$$b_s(o, v_*,) = \frac{c_s(o, v_*)}{\sum_v c_s(o, v)} \tag{4.2}$$

where, $c_s(o, v)$ is the number of instances of the pair (o, v) in the set, s. This measure can be used to quantify the bias for a object-verb pair in a dataset or for a model's prediction. For a dataset, $\mathcal{D}$, $c_{\mathcal{D}}(o, v)$ gives the number of instances of (o, v) pairs in it. Therefore, $b_{\mathcal{D}}$ represents the bias for the pair (o, v_*) in the dataset. A low value (≈ 0) of $b_{\mathcal{D}}$ means that the set is heavily biased against the pair while a high value (≈ 1) means that it is heavily biased towards the pair.

Similarly, we can define the bias of a model by considering the model's predictions as the dataset under consideration. For example, suppose that the model under consideration gives the predictions $\mathcal{P}$ for the dataset $\mathcal{D}$. We can define the model's bias as:

$$b_{\mathcal{P}}(o, v_*) = \frac{c_{\mathcal{P}}(o, v_*)}{\sum_v c_{\mathcal{P}}(o, v)} \tag{4.3}$$

where, $c_{\mathcal{P}}(o, v)$ gives the number of instances of the pair (o, v) in the set of the model's predictions $\mathcal{P}$.

A perfect model is one whose bias, $b_{\mathcal{P}}(o, v_*)$ is equal to the dataset bias $b_{\mathcal{D}}(o, v_*)$. However, due to bias amplification [12, 247], most models will have a higher/lower bias than the test dataset depending on the training set bias. That is, if the training set is heavily biased towards (resp. against) a pair, then the model's predictions will be more heavily biased towards (resp. against) that pair for the test set. The aim of a bias reduction method should be to bring the model's bias closer to the test set bias. Our experiments show that our proposed algorithm is able to

reduce the gap between the test set bias and the model prediction bias.

We consider a set of `(object,predicate)` pairs $\mathcal{Q} = \{(o_1, p_1), \ldots, (o_2, p_2)\}$. For each pair in $\mathcal{Q}$, we consider two scenarios: (1) the training set is heavily biased *against* the pair; (2) the training set is heavily biased *towards* the pair. For generating the training sets for a pair $q_i = \{o_i, p_i\} \in \mathcal{Q}$, for the first scenario, we remove all training samples containing the pair q_i and keep all other samples for the object. Similarly, for the second scenario, we remove all training samples containing o_i except those containing the pair q_i. For the pair, q_i the test set bias is b_i.

Given two models, the one with bias closer to test set bias is considered better. We show that our approach of augmenting the dataset brings the model bias closer to the test set bias. In particular, we consider $\mathcal{Q} = \{$`(horse,ride), (cup,hold)`$\}$, such that $b_1 = 0.275$ and $b_2 = 0.305$.

In the first scenario, baseline models trained on biased datasets have biases 0.124 and 0.184 for `(horse,ride)` and `(cup,hold)` respectively. Note that these are less than the test set biases because of the heavy bias against these pairs in their respective training sets. Next, we train models by augmenting the training sets using our methodology for only one neighbor of each object. Models trained on these new sets have biases 0.130 and 0.195. That is, our approach leads to a reduction in the bias *against* these pairs.

Similarly, for the second scenario, baseline models trained on the biased datasets have biases 0.498 and 0.513 for `(horse,ride)` and `(cup,hold)` respectively. Training models on datasets de-biased by our approach give biases 0.474 and 0.50. In this case, our approach leads to a reduction in the bias *towards* these pairs.

4.3.7 Visual Model

Our generalization module can be complementary to existing approaches. To illustrate this, we consider a simple visual module shown in Figure 4.6. It takes the union of b_h and b_o and crops the union box from the image. It passes the cropped union box through a CNN. The feature obtained, f_u is concatenated with f_h and f_o and passed through two FC layers. This module and the generalization module independently predict the probabilities for predicates and the final prediction is the average of the two. Using the generic object detector, the combined model gives an mAP of 15.82% on the Full HICO-Det dataset. This is better than the published best of 14.84%. This experiment shows that the generalization methodology proposed here is complementary to existing works which rely on purely visual data. Using our method in conjugation with other existing methods can lead to performance improvements.

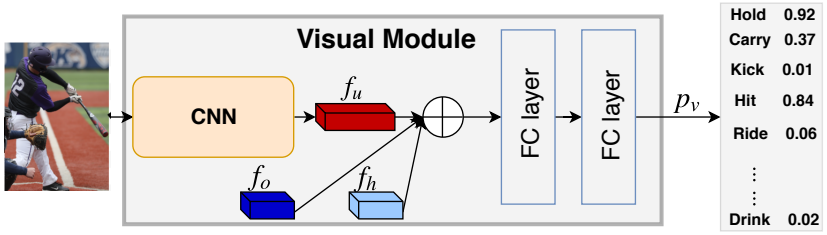


Figure 4.6: Simple visual module.

4.4 Discussion and Conclusion

4.4.1 Discussion

We discuss some limitations of the proposed approach. We have assumed that all predicates follow functional similarities. However, some predicates might only apply to particular objects. For example, you can `drag` a suitcase, but not a backback which is functionally similar to suitcase. Our model does not capture such constraints. Further work can focus on trying to explicitly incorporate such priors into the model. A related limitation of the proposed approach is the independence assumption on predicates. In fact, some predicates are completely dependent. For example, `straddle` usually implies `sit_on` for bicycles or horses. However, due to the inexhaustive labeling of the datasets, we (and most previous work) ignore this dependence. Approaches exploiting co-occurrences of predicates can help overcome this problem.

4.4.2 Conclusion

We have presented a way to enhance HOI detection by incorporating the commonsense idea that human-object interactions look similar for functionally similar objects. Our method can detect previously unseen (zero-shot) human-object relationships. We have provided experimental validation for our claims and have reported state-of-the-art results for the problem.

Chapter 5: Spatial Priming for Detecting Human-Object Interactions

Detecting human-object interactions involves localizing the interacting humans and objects and correctly predicting the type of interaction (predicate) between them. Humans can guess the type of interaction with just a quick glance at an image by considering the relative locations of the human and the object. For example, in Figure 5.1, the person on the left is very likely to be `sitting` on Chair-1 and `not interacting` with Chair-2. Similarly, the person in the middle is probably `dragging` the suitcase and the human on the right is `standing on` the snowboard and possibly `riding` it. This ability to use spatial relationships helps us in making guesses and eliminating improbable predictions. With additional visual information, we can refine these priors to give better predictions. This means that the relative spatial layout of the human and the object involved in a HOI scenario is greatly informative and should be exploited for predicting the interaction.

Current deep learning-based approaches either use a small hand-created feature [78] or binary maps called interaction patterns (IPs) [30]. Using hand-created features has the potential downside of not being able to encode the fine-grained spatial relationships among objects. This limitation can be overcome by using interaction patterns, which are binary maps representing the locations of the human

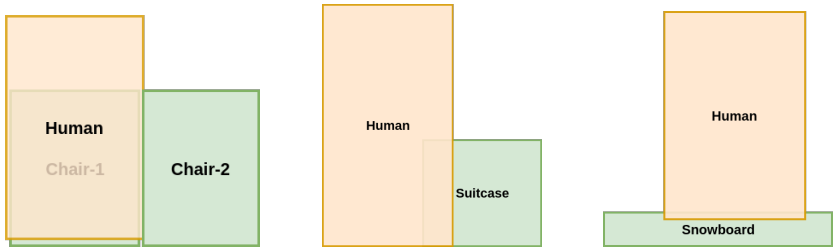


Figure 5.1: The relative spatial relationship between a human and an object provides much information about their interaction. We can infer that in the left image, the human is probably `sitting` on Chair-1 and `not interacting` with Chair-2. In the middle, the human might be `dragging` the suitcase. And the person on the right is probably `riding` a snowboard.

and the object in a HOI. Binary masks for the human and the object, as shown in Figure 5.1 can be useful for predicting a prior on the interaction. This can be refined by using more visual information from the image.

In this chapter, we build on this idea for HOI detection. We study the question: how can we utilize the spatial locations of the entities to improve HOI detection? Our proposed approach, described in Section 5.2, consists of a layout branch and a visual branch. The layout module outputs a prediction which is used as a prior by the visual module. This prior prediction primes the visual branch which then outputs the final predictions. Priming the visual module using predictions from the layout module enables our model to fully utilize the spatial layout of the human and the object. We treat the relative geometry of these entities as high-quality cues. This idea of priming the visual module is inspired by our interpretation of Kahneman's System-1 and System-2 formulation of human decision making [101]. The spatial layout module of our model is comparable to System-1 which is the "intuitive" part and the visual branch is comparable to System-2 which is the more "deliberative" part.

Our layout and visual modules share information at multiple stages. Such information sharing between different modalities [52] and at different levels of a network [128, 189] has been shown to make the models learn robust representations. Lateral connections provide a way to share information between modules processing different types of information. For example, [52] proposed lateral connections between motion and appearance branches for video action recognition. Our layout module receives information from the visual branch through lateral connections in the model. This sharing of information enables the layout module to make stronger predictions about the predicate.We put the proposed approach in context of prior work in Section 5.1.

We evaluate our proposed approaches on the challenging HICO-Det dataset [30]. In Section 5.3, we first present results for a simple baseline algorithm which uses a good object detector and already achieves state-of-the-art results for HOI detection. Our proposed model reaches a mean average precision (mAP) of 24.79% on the HICO-Det dataset, which is about 5.4 absolute points higher than current state-of-the-art. We also conduct extensive analysis of our proposed method to tease out the reasons for these improvements.

Finally, we discuss some limitations which are avenues for future research and conclude in Section 5.4.

The most important contributions of this work are: (1) propose spatial priming as a way to incorporate spatial layout of the human and object for HOI detection; (2) propose a model for HOI detection based on spatial priming and information sharing between a layout and a visual module. In addition, we conduct extensive

analysis and evaluation of the proposed model to isolate sources of performance improvement and report state-of-the-art results.

5.1 Related Work

Human-object interaction (HOI) prediction being a special and important subset of visual relationship prediction [137] is a well-studied problem. Early methods [43,74,234–236] for HOI prediction had mainly focused on developing hand-designed features and models. In particular, Yao et al. [235] proposed a random field model which encodes the idea that humans poses and objects can provide mutual context for each other. Delaitre et al. [40] built HOI features from spatial co-occurrences of body parts and objects. Hu et al. [88] used exemplars in the form of density functions representing an HOI. All of these are somewhat related to the proposed method owing to the use of the relative layout of humans and objects to reason about HOIs.

More recently, Mallya et al. [139] used CNN features from local and global context of a person along with a weighted loss to handle unbalanced training data. HO-RCNN [30] and InteractNet [69] employed separate human, object, and interaction streams for HOI prediction. In particular, [69] jointly learned human and object detectors along with HOI detectors. However, these methods did not leave any scope for zero-shot HOI prediction. In [227], Xu et al. utilized gaze and pose information through a gaze-driven context-aware branch. Gupta et al. [78] also used human-pose as fine-grained visual layout information. However, these methods re-

quire an additional model for predicting the pose. Unlike these, we argue that coarse relative layout along with the object identity provides sufficient cues to form a prior for interaction. We avoid the additional burden and potential errors of using a pose estimation model.

Several methods have utilized external **semantic knowledge for HOI prediction** [103, 160]. In this work, we too have used semantic information in the form of word vectors for object classes. These help transferring knowledge from an object to other similar object classes. Using semantic knowledge also helps in generalizing to zero-shot HOI categories. Zero-shot HOI detection has previously been studied in [187]. This followed several works on zero-shot object recognition [108, 222] and zero-shot object detection [21].

Like the proposed model, Li et al. [124] had also used priors for refining predictions. However, they learned "interactiveness priors" which only inform whether a human and an object are interacting or not. We, instead, add a prior which informs about the class of interaction based on the relative spatial layout of the human and object.

Spatial layout of the human and object is an important cue for predicting the interaction. Some prior works have tried to incorporate the spatial relationship by encoding it as a small hand-designed feature and passing it as input to a neural network [78]. Chao et al. [30] proposed "interaction patterns" for encoding the relative spatial location. Gao et al. [59] also used such interaction patterns as a branch of the model. However, neither of these models combined spatial layout as a prior for predicting HOI. It can be argued that even these methods considered

relative layout as secondary information to the visual features. In this chapter, we use IPs as binary spatial maps to represent the relative layout of the human and the object. We present a principled approach for exploiting the information contained in such spatial maps.

Lateral connections have been used previously for merging information from different spatial resolutions [128], for fusing optical and visual streams in two-stream networks for action recognition [52], and for fusing coarse and fine temporal resolutions for video recognition [51]. Shrivastava et al. [188] also used such lateral connections for priming an object detector by contextual information from semantic segmentation. Feature Pyramid Network (FPN) [128] used lateral connections for building high-level semantic feature maps for object detection. Similarly, Shrivastava et al. [189] proposed top-down modulations to incorporate finer details into an object detection architecture. Their bottom-up and top-down pathways are connected using lateral connections.

5.2 Approach

The proposed model is composed of a relative layout module (`L`) and a visual module (`V`) which share information at multiple stages. Predictions from `L` are used to prime `V` and the final prediction is the output of `V`. Figure 5.2 shows the entire pipeline of our approach. Our model takes detections from an object detector in the form of human-object pairs as input and outputs the probabilities for the predicates. We describe each component of our model next.

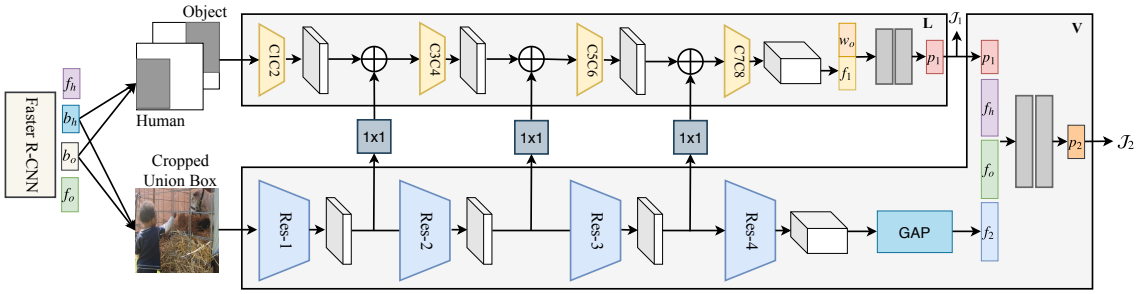


Figure 5.2: Proposed Pipeline. A Faster R-CNN object detector is used to detect humans and objects in an image. For each human-object pair, the interaction pattern, and union box are input to `L` and `V` respectively. Predictions from `L` are used for priming `V`. The visual module, `V` takes the union box, predictions from `L`, RoI pooled human and object features from the object detector and outputs the final probabilities over the predicates.

5.2.1 Object Detector

We start by using Faster R-CNN [175] to detect all humans and objects in an image and create all candidate human-object pairs. Each pair has an associated human bounding box, $b_h = \{b_h^{x_1}, b_h^{y_1}, b_h^{x_2}, b_h^{y_2}\}$ and an object bounding box, $b_o = \{b_o^{x_1}, b_o^{y_1}, b_o^{x_2}, b_o^{y_2}\}$, giving a union box:

$$b_u = \{\min(b_h^{x_1}, b_o^{x_1}), \min(b_h^{y_1}, b_o^{y_1}), \max(b_h^{x_2}, b_o^{x_2}), \max(b_h^{y_2}, b_o^{y_2})\} \tag{5.1}$$

We crop b_u from the image and use it as an input to the visual module `V`.

We generate binary spatial maps (interaction patterns) of the same size as b_u for the human and the object and stack these maps to produce a two-channel representation as shown in Figure 5.2. These spatial maps are used as inputs to the layout module `L`.

5.2.2 Layout Module

The spatial layout network, `L` is based on the idea that the relative layout and the semantic category of the object can provide sufficient cues to determine the prior probabilities for the interaction between a human and an object (see discussion in the introduction of this chapter). We use a shallow CNN as the layout network. This network takes the stacked spatial maps as input. We add features from the visual module (described next) to intermediate layers in `L` via 1×1 convolutions. This provides visual context about the human and the object. After the final convolution layer, we apply global average pooling to get the layout feature f_1.

Semantic knowledge. Only the relative spatial layout might not be enough to correctly determine the type of interaction. For example, in Figure 5.1, it becomes difficult to predict the relationship without the object identity. Therefore, we incorporate the object identity in `L`. To include semantic information about the object, we concatenate f_1 with the `word2vec` [147] representation of the object, w_o and pass the concatenated vector through two fully-connected layers which give the probabilities over all predicates. Using semantic information about the object also helps in improving the generalization of the model to interactions involving previously unseen objects (zero-shot detection). Using `word2vec` representations of the objects implicitly encodes semantic similarities between objects.

The output of the layout module are logits over the predicates which are used as inputs to the visual module. These logits act as a prior to the visual module which refines its output based on this spatial prior.

5.2.3 Visual Module

The visual module `V` uses the predictions from `L` along with the visual information from the cropped union bounding box to make the final prediction. We use a deeper network as the base network in `V`. As mentioned above, intermediate features from `V` are added to `L` to provide appearance and contextual information to the layout module. The base network in `V` provides a feature vector f_2 after global average pooling of the feature from the last convolution layer. We have two fully-connected layers at the end of the base convolution layers in `V`. As input to these layers, we concatenate the features f_2, the prior predictions from the layout module p_1, and the RoI-pooled human and object appearance features from the object detector, f_h and f_o. RoI-pooled features from the object detector provide explicit appearance information about the human and the object. The output of the visual module is the final output of the model.

5.2.4 Lateral Connections

We add features from intermediate layers in the visual module to intermediate layers in the layout module via 1×1 convolution layers. Adding the visual features to features in the layout module enables the model to explicitly share visual context not available the in layout module. Therefore, `L` can benefit from the spatial layout of the two interacting entities along with their appearances. This leads to a stronger spatial prior for the final stages of the visual module. We will empirically demonstrate the importance of using this layout information in Section 5.3.

5.2.5 Spatial Priming

Predictions from `L` prime the visual module based on the relative spatial layout of the human and the object. The layout module `L` provides strong priors for the predicate which are refined by the visual module which uses even more information about the human, object, and context appearance. Such priming enables the visual module to gain from the information contained in the relative spatial layout of the human and the object which is encoded by the binary spatial maps.

5.2.6 Training

We train `L` and `V` jointly. For both modules, we consider all predicates as independent and use a weighted binary-cross entropy loss. The weights are simply inversely proportional to the number of instances of the predicate in the dataset. The total loss is the sum of the two losses from `L` and `V`.

Note that, unlike many existing works, we predict the probabilities/confidence scores of each predicate for a human-object pair and not for the triplet `<human, predicate, object>` directly. This gives our method the ability to detect previously unseen HOI categories (zero-shot detection) and removes the limitations imposed by a particular dataset. To clarify, since we already have the object labels from the object detector, we only need to output the predicate in order to determine the interaction triplet. For example, the HICO-Det dataset contains 600 annotated HOI triplet categories of the form `<human,predicate,object>` but 117 predicates. In total, there are 9360 (117×80) possible interactions for this dataset. Only 600

of these are labeled. There might be more types of possible interactions than these. Our methods can potentially detect such unlabeled HOI categories too.

5.3 Experiments

We start with a brief description of the dataset and evaluation metrics and provide implementation details for our approach. We then discuss the model performance in fully-supervised and zero-shot settings. Finally, we discuss and analyze the model through extensive ablation studies.

5.3.1 Dataset and evaluation metrics

Following prior work, we use the challenging HICO-Det dataset [30] for evaluating our approach. This dataset contains 600 HOI triplet `<human, predicate, object>` categories involving 117 predicates and 80 objects. These categories are divided into: (a) Rare - 138 categories with less than 10 training samples, and (b) Non-rare categories. There are about $38,000$ training images containing about $120,000$ interactions and about $9,600$ test images with about $33,400$ HOIs.

Mean average precision (mAP) is used as the evaluation metric. A detected triplet is considered correct if both human and object overlaps (IoU) with the ground truth are greater than 0.5. Performance is reported for the full set of 600 classes and also for the rare and non-rare classes separately.

Due to its inconsequential size ($< 6,000$ training images and just 26 predicates), the V-COCO dataset [76] does not provide any new insights into HOI detec-

tion approaches. Unsurprisingly, most recent state-of-the-art methods [19, 78, 160] do not use V-COCO. However, for the sake of completeness, we evaluate our model on the V-COCO dataset as well.

5.3.2 Implementation Details

Following the state-of-the-art [19], we start by fine-tuning a ResNet-101 [85] based Faster R-CNN [175] object detector for the HICO-Det dataset [30]. The detector was originally trained on the COCO dataset [130] which has the same 80 object classes. Fine-tuning enables the detector to confidently detect objects more likely to be involved in an interaction. This helps in improving the performance of downstream predicate classifiers. Please see supplementary materials for details.

To create the training dataset, we consider all detections for which the detection confidence is greater than 0.75 and the overlap with a ground-truth human or object box is greater than 0.7. We create human-object pairs for each image using these detections and end up with about $250,000$ training HOI triplets. For test proposals, we select only those object and human proposals which have a confidence score greater than 0.9 for a particular class. This ensures that we get only high confidence object detections and make fewer errors because of incorrectly detected objects and humans. Each detection has an associated feature vector and bounding boxes. We use the human and object bounding boxes, b_h and b_o respectively to compute the union box and the binary spatial maps.

For our model, the visual module is a ResNet-50 network and the layout mod-

Table 5.1: **Architecture of `L`**. C1-C8 are convolution layers. Layer dimensions are in the shape `kernel_width` × `kernel_height` × `output_channels`. Numbers in parenthesis are strides.

Layer	Layer Dimensions	Output Sizes
C1C2	7×7×64 (2), MaxPool (2), 3×3×256 (1)	56×56×256
C3C4	1×1×128 (1), 3×3×512 (2)	28×28×512
C5C6	1×1×256 (1), 3×3×1024 (2)	14×14×1024
C7C8	1×1×512 (1), 3×3×2048 (2)	7×7×2048
GAP	7×7	1×1×2048
FC1	1024	1024
FC2	512	512

ule is a shallow 8-layer CNN with the layers described in Table 5.1. Each layer of `L` contains a ReLU non-linearity and batch-normalization. We add lateral connections from each Residual block in `V` to `L`, i.e., there are three lateral connections. Features from the residual blocks, Res-1, Res-2, and Res-3 are added to the respective places in `L` as shown in Figure 5.2. The fully connected layers are of sizes 1024 and 512 in both `L` and `V`. We reiterate that both `L` and `V` give the probabilities for the 117 predicates. This is unlike many previous methods which directly predict the HOI triplets (600 categories). We use the object labels from the object detector to output the final triplet. This also enables us to detect previously unseen HOIs (zero-shot detection).

In all our experiments, we train the model for 10 epochs with an initial learning rate of 0.1 which is dropped by a tenth every 3 epochs. Note that the object detector and the semantic word-vectors are frozen while training our models, i.e., the detector needs to be trained only once.

Table 5.2: Baseline results (mAP %)

Method	**Full** (600 classes)	**Rare** (138 classes)	**Non-rare** (462 classes)
Baseline ResNet-50	20.80	15.63	22.34
Baseline ResNet-50+$f_h + f_o$	21.49	14.43	23.60

5.3.3 Results

Strong Baseline. We start with a baseline CNN which predicts the predicates just based on the cropped union box. We first use a ResNet-50 (R-50) network as the classifier which takes a cropped union box as input and outputs the probabilities for each predicate. This network achieves an mAP of 20.80% for the HICO-Det test set. This is a strong albeit simple baseline which is already better than the current state-of-the-art performance of 19.40% (Table 5.3). This reveals that the existing methods can benefit from simplifying the algorithm and just using a better object detector and a stronger feature extractor. A simple model like classifying the union box obtained from detections from an object detector is enough to achieve better performance than existing methods. Adopting the common practice [69, 124] of using the features from the object detector, we append the RoI-pooled features to the features from the R-50, and obtain an mAP of 21.49%. We summarize these results in Table 5.2.

Comparison with Prior Work. We compare the performance of our model with past work in Table 5.3. Our model achieves an mAP of 24.79%, which is over 2.8 absolute percentage points higher than Functional Generalization (Chapter 4) on

Table 5.3: Comparison with prior work. The performance (mAP %) obtained by our method is significantly higher than existing methods.

Method	**Full** (600 classes)	**Rare** (138 classes)	**Non-rare** (462 classes)
Shen *et al.* [187]	6.46	4.24	7.12
HO-RCNN + IP [30]	7.30	4.68	8.08
HO-RCNN + IP + S [30]	7.81	5.37	8.54
InteractNet [69]	9.94	7.16	10.77
GPNN [162]	13.11	9.34	14.23
iHOI [227]	13.39	9.51	14.55
Xu *et al.* [228]	14.70	13.26	15.13
ICAN [59]	14.84	10.45	16.15
Wang *et al.* [213]	16.24	11.16	17.75
Gupta *et al.* [78]	17.18	12.17	18.68
Interactiveness Prior [124]	17.22	13.51	18.32
RPNN [249]	17.35	12.78	18.71
PMFNet [208]	17.46	15.65	18.00
Peyre *et al.* [160]	19.40	15.40	20.75
Functional Gen. (Chapter 4)	21.96	**16.43**	23.62
Ours	**24.79**	14.77	**27.79**

Table 5.4: Zero-shot HOI detection (mAP %).

Method	**Unseen** (120 classes)	**Seen** (480 classes)	**All** (600 classes)
Shen *et al.* [187]	5.62	-	6.26
Functional Gen. (Chapter 4)	10.93	12.60	12.26
Ours	**11.06**	**21.41**	**19.34**

the Full set of the HICO-Det dataset. Our method also performs about 4.2 absolute percentage points better on Non-rare classes. Interestingly, at the same time, even though we do not target them explicitly, our model achieves competitive performance on Rare classes too. Note that the methods in [19] and [160] are explicitly designed to target rare and unseen classes.

We also point out that, even using the original COCO detector instead of our fine-tuned detector, our model achieves an mAP of 19.45%. This is the highest among all methods using an object detector trained on COCO. In particular, the mAP achieved by the proposed method is significantly higher ($2-12\%$ mAP) than previous methods [30, 59, 213] which aim to utilize the relative spatial layout of the two entities. In addition, we obtain a higher performance than RPNN [249] and PMFNet [208] which use additional pose information using models trained on large datasets. This demonstrates the strength of Spatial Priming as a way of modeling the geometric layout.

Zero-Shot HOI Detection. The proposed approach can help improve the performance for zero-shot HOI detection. Table 5.4 compares the performance of our method with the state-of-the-art methods [19,187] on zero-shot HOI detection. Prior

work divides the classes into a set of 120 unseen and 480 seen classes. We use the same setting here. The model is trained with training data for only the seen classes and is evaluated on the set of unseen classes. Note that the classes are divided such that the there is at least one interaction involving each of the 80 objects in the training set, i.e., the model is trained with at least one HOI involving each object. From Table 5.4 we observe that our model achieves a higher mAP than Functional Generalization for Unseen classes while also improving the mAP for Seen classes by a huge margin. We have used the same train-test splits as [19].

5.3.4 Ablation Analysis

We now extensively analyze our model in Tables 5.5, 5.6, and 5.7.

Importance of w_o. Word-vectors w_o encode the semantic similarities between objects. Table 5.5a shows that using the word-vector in the layout module leads to performance improvement. The complete model which uses the `word2vec` vectors w_o achieves an mAP 24.79%. Removing this word-vector leads to a lower performance (24.47%).

Type of Lateral Connection. Table 5.5b illustrates that adding the features from the visual module to the geometric module achieves higher performance than concatenating the features. The mAP in the case of addition of features is 24.79%. Compare this to the mAP of 24% when the features are concatenated instead. The reason for this is that adding features from the visual module forces `L` to explicitly focus on the human and object. This ensures that the relevant regions of the image

Table 5.5: Ablation studies for the model. We report mAP (%) in each case. In all sub-tables "Standard" refers to the model shown in Figure 5.2.

(a) **Effect of $\mathbf{w_o}$.** Row 2 is the standard model without w_o. Note that the performance without w_o is lower than the Standard case. This is particularly true for the Rare classes.

Setting	Full	Rare	Non-rare
Standard	**24.79**	**14.77**	27.79
Standard - w_o	24.47	12.16	**28.14**

(b) **Lateral connection methods.** Concat is the model with lateral additions replaced by concatenation. 3x3add uses 3×3 convs in lateral connections instead of 1×1 used in the Standard setting.

Setting	Full	Rare	Non-rare
Standard	**24.79**	**14.77**	**27.79**
Concat	24.00	13.91	27.02
3×3add	24.21	13.34	27.47

(c) **Importance of `L`.** ImgCNNA-ImgR50 is the model where input to `L` is the cropped union box. Similarly, in ImgR50-ImgR50, the layout module is a ResNet-50 with the union box as input. (Standard is IPCNNA-ImgR50)

Setting	Full	Rare	Non-rare
Standard	**24.79**	**14.77**	**27.79**
ImgCNNA-ImgR50	22.28	10.87	25.69
ImgR50-ImgR50	24.07	11.96	27.68

(d) **Utility of $\mathbf{f_h}$, $\mathbf{f_o}$.** Concat contains concatenated lateral connections. Standard$-f_h-f_o$ has no f_h and f_o. Standard-Larger contains larger hidden layers and no f_h and f_o. Similarly for Concat$-f_h-f_o$ and Concat-Larger.

Setting	Full	Rare	Non-rare
Standard	**24.79**	**14.77**	27.79
Standard$-f_h-f_o$	22.32	13.14	25.07
Standard-Larger	24.60	13.58	**27.89**
Concat	24.00	13.91	27.02
Concat$-f_h-f_o$	21.87	13.05	24.51
Concat-Larger	23.41	14.44	26.09

(e) **Different lateral connections.** Conn1 is the model with just one lateral connection from `V` to `L` which is at Res-1. Conn2 has just one lateral connection at Res-2 and Conn3 has the lateral connection at Res-3. `L-V` has all connections from `L` to `V`.

Setting	Full	Rare	Non-rare
Standard	**24.79**	**14.77**	**27.79**
L-V	23.81	13.44	26.91
Conn1	23.63	10.75	27.48
Conn2	24.01	12.86	27.34
Conn3	22.87	11.42	26.28

are given more importance. Similarly, when using 3×3 convolutions in the lateral connections instead of 1×1, the performance is slightly lower. This is because using 3×3 convolutions increase the receptive field of the features. This dilutes the focus on the human and the object which in turn leads to a lower performance.

Layout Module. The importance of using the relative spatial layout of the human and object is demonstrated using the data in Table 5.5c. The first row in the table is the standard case when the human and object spatial maps are given as input

to the shallow layout network. This model gives an mAP of 24.79%. Now, if we remove the binary spatial maps (IP) and input the cropped union box image to the layout module too, the performance of the model drops to just 22.28% (second row). Note that this model has the same number of parameters as the previous model. The only difference is the input to `L`. To ensure that the drop in performance is not due to a weak layout network, we replace the small CNN in the layout module with a ResNet-50 network. Again, the input to both the layout and visual branches is the cropped union box. Even this model, with a much larger number of parameters than the standard case, gives an mAP of just 24.07%. This shows that relative spatial layout of the human and the object provides irreplaceable information for determining the type of interaction.

Importance of f_h and f_o. From Table 5.5d, we observe that a model gives a lower performance if appearance features from the object detector are not used. For example, the Standard model reaches an mAP of 24.79% while the Standard model trained without f_h and f_o reaches only 22.32%. Similarly, the performance for the Concat model (from Table 5.5b) goes down from 24% to just 21.87% on removing the features. Clearly, f_h, and f_o help in achieving higher performance. Recall that we had observed the same effect with the Baseline model (Table 5.2).

To analyze if these improvements are because of a larger number of parameters, we removed f_h and f_o and increase the sizes of the fully connected layers such that the number of trainable parameters in this model and the Standard model are roughly the same. We call this model Standard-Larger. This model gives an mAP

of 24.60%. This is down from 24.79% obtained by the Standard model. Similarly, the Concat-Larger model gives an mAP of 23.41%, down from the Concat model which gave 24%. So, even though some of the performance gain when using the appearance features could be due to a larger number of parameters, it does not explain the whole difference. We believe that the features f_h and f_o do, in fact, provide useful information for predicting the HOI.

Different Connections. We show that having lateral connections at multiple depths in the network is important for obtaining a good performance. We study whether having just one lateral connection can be enough. From Table 5.5e, we infer that the answer is *no*. Just one lateral connection after either Res-1 block, Res-2 block, and Res-3 block (rows 3, 4, and 5 respectively) gives worse performance than having connections at all three places. In particular, having just one connection after Res-3 gives the lowest performance. This is because by this depth the visual module loses most spatial information and the layout module does not benefit from adding visual features. This shows that frequent information sharing between the two modules via lateral connections gives significant performance improvements. We also observe that passing information from the spatial layout module to the visual module also achieves a lower mAP.

To analyze the effect of each of the component in our model in more detail, we conduct ablation studies in two further settings. First, we study the utility and behaviour of lateral connections without priming. We remove the loss $\mathcal{J}_1$ and instead of adding layout priors, p_1 to the visual module, we directly add the global

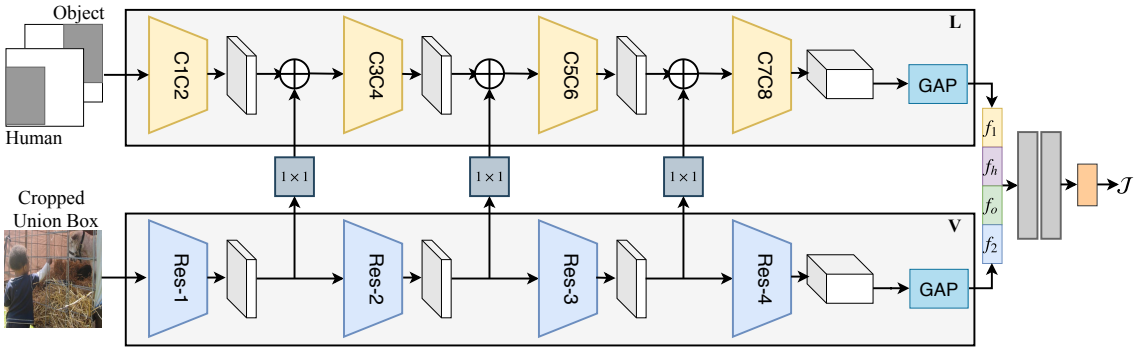


Figure 5.3: **No priming (NP)**. This model removes the spatial priming from our model (Figure 5.2). Human and object bounding boxes from an object detector give the interaction patterns and the union box. Global Average Pooled (GAP) features from the geometry and visual networks are concatenated to the human and object RoI pooled features from the object detector. Two FC layers are used to get the probabilities/confidences over the predicates.

Table 5.6: **NP Results** (mAP %). NP is the model shown in Figure 5.3 with lateral connections from `V` to `L`. NC is the same model without lateral connections. Similarly, `L-V` has connections from the layout branch to the visual branch. `V-L-concat` concatenates the features from `V` and `L` instead of adding.

Method	**Full** (600 classes)	**Rare** (138 classes)	**Non-rare** (462 classes)
`V-L-add` (NP)	**23.41**	12.14	**26.78**
NC	22.56	**12.78**	25.48
`L-V`	22.45	12.23	25.50
`V-L-concat`	22.76	11.78	26.04

average pooled features, f_1. This gives the model shown in Figure 5.3. We call this model NP.

No Priming. The first row in Table 5.6 gives the performance of the NP model (no priming) shown in Figure 5.3. This model achieves an mAP of 23.41% on the Full HICO-Det dataset. Notice that this is higher than the Baseline model discussed earlier (21.49% Table 5.2). This highlights the importance of the spatial layout even in this simpler setting.

We further analyze the behavior of this model in different conditions. In Table 5.6 `V-L` is the model with lateral connections from the visual module (`V`) to the layout module (`L`). To illustrate the positive impact of these lateral connections, we remove all lateral connections and train the resulting model. This model is called "NC" (no connection) in Table 5.6. NC reaches an mAP of only 22.56%. Clearly, lateral connections enable better utilization of the relative spatial layout of a person and an object. However, note that this is still higher than the Baseline model (21.49%), clearly demonstrating that leveraging layout information is important for improving HOI detection performance.

Further, we observe that connections from layout module to the visual module (`L-V`) give almost the same performance as having no connection (NC). Also, the final row in Table 5.6 is the case where we concatenate the intermediate features from the visual module to the features of the layout module instead of adding. This model, though better than having no connections, is still worse than the `V-L` model. We believe that in the case of concatenation, the network does not learn to attend to the human and the object. On the other hand, when we add the features instead, we explicitly force the network to attend to the human and object regions of the image. This enables it to learn better mappings from human and object appearances to the correct predicate. Recall that we had seen similar behavior in Table 5.5b.

Next, we study the effect of removing lateral connections from our model. This is an important case and will show how informative the layout module is on its own. This will help us pinpoint how the components of our model behave in

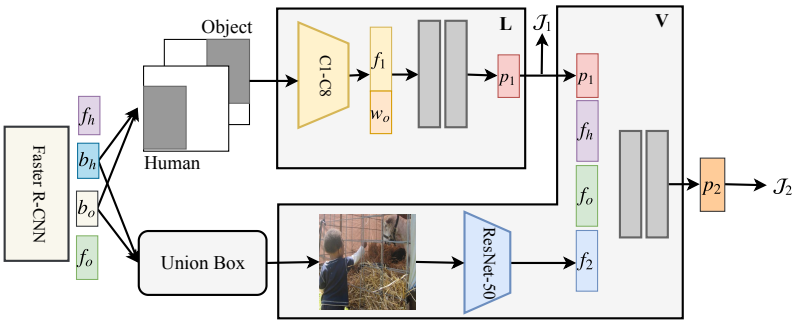


Figure 5.4: **No lateral connections (NL)**. We remove lateral connections from our model (Figure 5.2). Now, `L` predicts the interaction just based on the spatial layout. These predictions are given as a prior to `V` which also uses the union bounding box and the RoI pooled features from the object detector to make the final prediction.

the presence of only spatial priming without lateral connections. We remove the lateral connections from our model in Figure 5.2. This gives us the model shown in Figure 5.4. We call this model NL.

No Lateral Connections. Results and ablation studies for the NL model are listed in Table 5.7. The best model reaches 23.90% in mAP on the HICO-Det dataset. It contains a shallow layout branch `L` which predicts the predicate based only on the spatial layout of the human and the object. This prediction is used as a prior by the visual network `V` which gives the final prediction. We highlight the performance of the layout network `L`. It achieves an mAP of 18.35% on the Full set of HICO-Det. This shows that there is significant information about the interaction category contained in the relative spatial layout of the human and object. When properly trained, using only this information might be better than most existing methods (13/15 methods in Table 5.3).

Again, the importance of a layout-based prior is apparent when comparing the performance of this model with the performance of the baseline R-50 model

Table 5.7: **NL Results** (mAP %). First row (NL) is the model shown in Figure 5.4. NL - f_h - f_o represents the model trained without the appearance features from the object detector. NL - w_o is NL without the word vector for the object.

Method	Model	**Full** (600 classes)	**Rare** (138 classes)	**Non-rare** (462 classes)
NL	L	18.35	8.20	21.38
	V	**23.90**	10.82	**27.81**
NL - f_h - f_o	L	17.44	10.14	19.62
	V	23.19	**14.71**	25.72
NL - w_o	L	16.33	8.45	18.69
	V	22.91	11.29	26.39

which had reached only 21.49% (Table 5.2). The last row in Table 5.7 shows that removing the word vector w_o from the model leads to a drop in performance. This is driven down by the reduction in the performance of the layout model L which went from 18.35% in the usual case to just 16.33%. Removing the appearance features from the detector, f_h and f_o, also results in lower performance. We had seen the same trends even in the presence of lateral connections in Table 5.5. Also, note that the performance for NL (23.90%) is higher than the performance for NP (23.41% Table 5.6), showing that the idea of spatial priming is a significant source of improvement achieved by our proposed model.

5.3.5 Experiments on V-COCO

Similar to prior methods [69,208,249] we use the 24 action classes involving a person and an object. We use our ResNet-101 object detector to extract the human and object bounding boxes from each image. For generating the training set, we use all

Table 5.8: Comparison with prior work for the V-COCO dataset. The performance (mAP_{role} %) obtained by our method is higher than existing methods.

Method	**mAP_{role}**
Gupta et al. [76]	31.8
InteractNet [69]	40.0
GPNN [162]	44.0
ICAN [59]	45.3
RPNN [249]	47.5
Wan et al. [208]	48.6
$RP_{T2}C_D$ [124]	48.7
Spatial Priming (Ours)	**49.2**

proposals which overlap with a ground-truth entity box with an IoU greater than 0.5. For testing, we use human and object proposals with confidence > 0.8. A major difference between the HICO-Det dataset [30] and V-COCO is the absence of annotations for the `no-interaction` or `background` class. We generate samples for `no-interaction` by considering un-labeled human-object interactions as belonging to this class. Following the standard practice in object detection [175], we use the background and labeled classes in a ratio of 3:1.

Table 5.8 shows that the performance achieved by Spatial Priming (49.2% mAP) is significantly higher than most existing methods. In Table 5.9, we list the class-wise AP obtained by our method for the 24 classes under consideration.

5.4 Discussion and Conclusion

We discuss some limitations of the approach which can be avenues for further improvements and finally conclude.

Table 5.9: Class-wise AP obtained by our Spatial Priming approach for the V-COCO dataset.

Class	**AP$_{\text{role}}$**
hold-obj	36.81
sit-instr	32.94
ride-instr	64.34
look-obj	42.34
hit-instr	69.02
hit-obj	39.39
eat-obj	46.70
eat-instr	13.31
jump-instr	52.69
lay-instr	31.66
talk_on_phone-instr	30.15
carry-obj	37.45
throw-obj	41.72
catch-obj	52.94
cut-instr	34.04
cut-obj	48.30
work_on_computer-instr	64.85
ski-instr	46.03
surf-instr	76.43
skateboard-instr	86.68
drink-instr	45.01
kick-obj	78.37
read-obj	33.18
snowboard-inst	75.36
Average Role AP	49.15

5.4.1 Discussion

In this chapter, we have not explicitly considered ways of improving detection for rare classes. The competitive performance for rare classes in Table 5.3 is a by-product of our approach, particularly, using semantic knowledge in the form of `word2vec`

representations. HOI datasets will always suffer from the long-tail problem. Future research should focus on improving performance for rare classes.

5.4.2 Conclusion

We have presented an approach for using the relative layout information of a human and an object for detecting the interactions between them. Our proposed model consists of two modules: one for processing the relative spatial layout of a human and an object, and the other for processing visual information. The visual module is primed using the prediction of the layout module. We have systematically analyzed the model and our experiments shown that this method can significantly out-perform state-of-the-art methods for HOI detection.

Chapter 6: Image-Set Visual Question Answering

Answering natural-language questions about images requires understanding both linguistic and visual data. Since its introduction [14], Visual Question Answering has attracted significant attention. Several related datasets [14, 93, 140, 239, 254] and many methods [60, 71, 131, 153, 233] have been proposed since.

In this work, we introduce the new task of Image Set Visual Question Answering (ISVQA). It aims to answer a given free-form natural-language question based on a small set of images. The proposed ISVQA task could require reasoning over objects and concepts in different images to predict the correct answer. For example, for Figure 6.1, a model has to find the relationship between the `bed` in the top-left image and the `mirror` in the top-right, via `pillows` which are common to both the images. This example shows the unique challenges associated with image-set VQA. A model for solving this type of problems has to understand the question, find the connections between the images, and use those connections to relate objects across images. Similarly, in Figure 6.2, the model has to avoid double-counting recurring objects in multiple images. These challenges associated with scene understanding have not been explored in existing single-image VQA settings but frequently happen in the real world. Humans require limited effort to solve them but they are difficult

Figure 6.1: Given the set of images above, and the question "What is hanging above the bed?", in order to answer the question, it is necessary to connect the bed in the top-left image to the mirror in the top-right image. To answer this question a model needs to understand the concepts of "bed", "mirror", "above", "hanging", etc. and be able to relate the bed in the first image with the headrest and pillows in the third image.

for machines.

Instances of the ISVQA task include answering questions about images taken at different times (e.g. videos or sequential images taken at several times like in camera trap photography), at different locations (e.g. multiple camera streams from indoor or outdoor locations), or from different viewpoints (e.g. live sports coverage, multiple views of objects). Some of these settings contain images taken from the same scene, while others might involve images of a larger span. In this work, we focus on the setting where the images are taken from different locations or viewpoints in the same scene.

In this setting, ISVQA may require finding the same objects in different images and determining the relationships between different objects within or across images. It can also entail determining which image/images are the most relevant for the

Figure 6.2: When asked the question "How many rectangles are on the interior doors?", the model should be able to provide the ground-truth answer ("four") and avoid counting the rectangles multiple times even though they occur in multiple images.

question and answering the question based only on them, ignoring the other images. Along with the language-based question, ISVQA asks for solutions to two research challenges: a) How to use natural language to guide scene understanding across multiple views/images; and b) how to fuse information from relevant images to reason about relationships among entities.

To enable research into these problems, we built two datasets for ISVQA - one for indoor scenes and the other for outdoor scenes. The indoor scenes dataset comes from Gibson Environment [220] and contains 91,479 human-generated questions, each for a set of images - for a total of 48,138 image sets. Similarly, the outdoor scenes dataset comprises of 49,617 questions for 12,746 image sets. The images in the outdoor scenes dataset come from the nuScenes dataset [25]. We introduce the datasets, explain the data collection methodology, and the statistics of the datasets in Section 6.3.

The indoor scenes ISVQA dataset contains two parts: 1.) Gibson-Room; and 2.) Gibson-Building. This is to facilitate spatial and semantic reasoning both in a localized region and an extended area in the same scene. The outdoor scenes dataset contains image sets taken from mostly urban environments.

We propose two extensions of single-image VQA methods as the baseline approaches to investigate the ISVQA task and the datasets. Such baselines meet significant difficulties in solving the ISVQA problem, and they reflect the particular challenges of the ISVQA task. We also present the statistics of the datasets, by analyzing the types of question, distributions of answers for different types of questions, and biases present in the dataset.

In summary, we make the following contributions:

- propose ISVQA - Image Set Visual Question Answering as a new setting for scene understanding via question answering;
- introduce two large-scale datasets for targeting the ISVQA problem. In total, these datasets contain 141,096 questions for 60,884 sets of images. Each question has at least three annotations of answers.
- establish baseline methods on ISVQA tasks to recognize the challenges and encourage future research.

6.1 Related Works

Visual Question Answering settings. The basic free-form open-ended VQA setting was proposed in [14] and involves answering natural language questions about

images. This setting was further extended by the VQA2.0 dataset [71] which is a balanced and extended version of the original VQA dataset. The VisualGenome dataset [112] also contains annotations for visual question-answer pairs at both image and region levels. Visual7W [254] built upon the basic VQA setting and introduced visual grounding to VQA. This enabled inclusion of visual answers in addition to textual answers for the VQA questions. Several other VQA settings target specific problems or applications. For example, VizWiz [80] was designed to help develop algorithms which can answer questions asked by people who are blind. RecipeQA [229] is targeted for answering questions about recipes from multi-modal cues. OK-VQA [140] targets questions which require external knowledge in addition to the images. TallyQA [7], and HowMany-QA [203] specifically target counting questions for single images. In addition to these works, which use real images for VQA, the CLEVR [98] benchmark and dataset uses synthetically generated images of rendered 3D shapes and is aimed towards understanding the geometric relationships between objects. IQA [70] is also a synthetic setting where an agent is required to navigate a scene and reach the desired location in order to answer the question.

Unlike existing work, the proposed ISVQA setting targets scene understanding and comprises of questions which might require multiple images to answer. This important setting has not been studied before and necessitates a specialized dataset. Additionally, answering almost every question in our dataset requires a model to ignore some of the images in the set. This capability is mostly absent from many state-of-the-art VQA models. A part of ISVQA also comprises of images rendered

from 3D scans of houses and offices and contains questions which require geometric reasoning across images. This requires a model to develop an understanding of the scene.

We also distinguish our work from the video VQA setting. Unlike many such datasets (e.g. TVQA [120], TVQA+ [121], MovieQA [201]) which also contain scripts or subtitles, our dataset does not contain any textual cues. Also, videos are temporally continuous and are mostly taken from a stationary view-point. This makes finding associations between objects across frames easy, even for datasets which do not provide textual cues (e.g. tGIF-QA [96]).

VQA methods. Several recent methods have achieved excellent performance for VQA tasks. Most of these methods use some kind of attention mechanisms to focus on the regions in an image which are most relevant to the question. For example, [233] proposed stacked attention networks which use question features as queries to find the most relevant image regions in several stacked steps. Similarly, [11] proposed a bottom-up and top-down attention mechanism for answering visual questions.

In addition to such methods, several methods which use co-attention (or bi-directional) attention over questions and images have been proposed. Such methods include [60, 138, 150, 212], all of which use the information from one modality (text or image) to attend to the other. Somewhat different from these is the work from Gao et al. [61] which proposed the multi-modality latent interaction module which can model the relationships between visual and language summaries in the form of latent vectors.

Unlike these, [44] used reasoning modules over detected objects to answer visual questions about geometric relationships between objects. Similarly, Santoro et al. [182] proposed using Relation Networks to solve specific relational reasoning problems. Neither of these approaches used attention mechanisms. Though these are interesting and relevant directions of work, in this dissertation, we mostly focus on attention-based mechanisms to design the baseline models.

6.2 ISVQA Problem Formulation and Baselines

We start by formally describing the problem and then introduce the baseline methods.

6.2.1 Problem Definition

Refer to Figure 6.3 for some examples of the ISVQA setting. Given a set of images, $S = \{I_1, I_2, \ldots, I_n\}$, and a natural language question, $Q = \{v_1, v_2, \ldots, v_T\}$, where v_i is the i^{th} word in the question, the task is to provide an answer, $a = f(S, Q)$, which is true for the given question and image set. The function f can either output a probability distribution over a pre-defined set of possible answers, $\mathcal{A}$, or select the best answer from several choices which are input along with the question, i.e., $a = f(S, Q, C_Q)$, where C_Q is the list of choices associated with Q. The former is usually called open-ended QA and the latter is called multiple-choice QA. In this work, we mainly deal with the open-ended setting. Another possible setting is to actually generate the answer using a text generation method similar to image-

Figure 6.3: Some examples from our dataset which demonstrate the ISVQA problem setting. In each case, the input is a set of images and a natural language question. A model designed to solve ISVQA needs to correctly answer the question based on the given set of images.

captioning. But, most existing VQA works focus on either of the first two settings and therefore, we also consider the open-ended setting in this work. We leave the harder problem of generating answers to future work.

6.2.2 Model Definitions

Now, we describe some baselines for the ISVQA problem. These baselines directly adapt single image VQA models. The first of these processes each image separately and concatenates the features obtained from each image to predict the answer. The second baseline directly adapts VQA methods by simply stitching the images and using single image VQA methods to predict the answer.

6.2.2.1 Concatenate-Feature Baseline

Starting from a given set of n images $S = \{I_1, I_2, \ldots, I_n\}$, we use a region proposal network (RPN) [175] to extract region proposals $R_i, i = 1, 2, \ldots, n$ and the corresponding RoI-pooled features (fc6). With some abuse of notation, we denote the region features obtained from each image as $R_i \in \mathbb{R}^{p \times d}, i = 1, 2, \ldots, n$, where p is

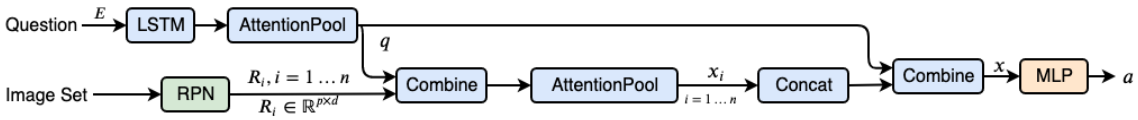


Figure 6.4: **Concatenate-Feature Baseline**. This method adapts a single-image VQA model to an image set $S = \{I_1 \ldots I_n\}$. We first extract region proposals, R_i from each image I_i. The model attends over the regions in each image separately using the question embedding q. Pooling the region features gives a representation of an image as x_i. These are concatenated and combined (element-wise multiplied) by the question embedding to give the joint scene representation x. We use fully-connected layers to predict the final answer.

the number of region features obtained from each image and d is the dimension of the features. We are also given a natural language question $Q = \{v_1, v_2, \ldots, v_T\}$, where v_i is the i^{th} word, encoded as a one-hot vector over a fixed vocabulary V of size d_V. For all the models, we first obtain question token embeddings $E = \{W_w^T v_i\}_{i=1}^T$, where $W_w \in \mathbb{R}^{d_V \times d_q}$ is a continuous word-vector embedding matrix. We obtain the question embedding feature using an LSTM-attention module, i.e., $q = \text{AttentionPool}(\text{LSTM}(E)) \in \mathbb{R}^{d_q}$.

Figure 6.4 shows an outline of the model. For each image, I_i, we obtain the image embedding, x_i by attending over the corresponding region features R_i using the question embedding q.

$$x_i = \text{AttentionPool}(\text{Combine}(R_i, q)) \tag{6.1}$$

where, we use element-wise multiplication (after projecting to suitable dimensions) as the Combine layer and AttentionPool is a combination of an Attention module over the region features which is calculated through a softmax operation and a Pool

operation. The region features are multiplied by the attention and added to obtain the pooled image representation. For a single image, this model is an adaptation of the recent Pythia model [192] without its OCR functionality. We concatenate the image features x_i and element-wise multiply by the question embedding to obtain the joint embedding

$$x = \text{Combine}(\text{Concat}(x_1, x_2, \ldots, x_n), q) \tag{6.2}$$

where the Combine layer is again an element-wise multiplication. This is passed through a small MLP to obtain the distribution over answers, $P_A = \text{MLP}(x)$.

6.2.2.2 Stitched Image Baseline

Our next baseline is also an adaptation of existing single-image VQA methods. We start by stitching all the images in an image set into a mosaic, similar to the ones shown in Figure 6.3. Note that the ISVQA setting does not require the images in an image set to follow an order. Therefore, the stitched image obtained need not be panoramic. We train the recent Pythia [192] model on the stitched images and report performance in Table 6.2.

Using the two baselines, we will demonstrate that ISVQA is not a trivial extension of VQA. Solving ISVQA requires development of specialized methods. Even high-performing VQA models perform poorly on ISVQA.

6.2.2.3 Evaluating Biases in the Datasets

In addition to the two baselines mentioned above, we also evaluate the following prior-based baselines to reveal and understand the biases present in the datasets.

Naïve Baseline. The model always predicts the most frequent answer from the training set. For nuScenes, it always predicts "yes", while for Gibson it predicts "white". Ideally, this should set a minimum performance bar.

Hasty-Student Baseline. In this baseline, a model simply finds the most frequent answer for each type of question. In this case, we define a "question type" as the first two words of a question. For example, a hasty-student might always answer "one" for all "How many" questions. This is similar to the hasty-student baseline used in [126] (MovieQA).

Question-Only Baseline. In this model, we ignore the visual information and only use question text to train a model. Our implementation takes as input only the question embedding, q which is passed through several fully-connected layers to predict the answer distribution. This baseline is meant to reveal the language-bias present in the dataset.

6.3 Dataset

In this section, we describe the ISVQA datasets – the annotation procedure, the challenges associated with getting annotations, and the dataset statistics.

The main goal of our data collection effort is to aid multi-image scene un-

derstanding via question answering. We want to focus on both indoor and outdoor scenes. Therefore, we use two publicly available datasets (Gibson [220] and nuScenes [25]) as images-sources for our datasets. We select these datasets because they represent diverse settings both indoors and outdoors, and sets of images can be obtained from them to represent scenes.

The indoor images in our indoor-scene dataset are obtained from the Gibson environment. We use the Habitat API [183] to extract view-points from an indoor scene. On the other hand, we use the nuScenes dataset for the outdoor scenes dataset. It contains sets of images which represent the 360 degree field of view from an outdoor urban scene. The images in this dataset are taken from cameras on a self-driving car.

6.3.1 Annotation Collection

We now describe the methodology used for generating and annotating both paths of the dataset in detail.

6.3.1.1 Indoor Scenes

The Gibson dataset and environment [220] is a collection of 3-dimensional scans of indoor spaces, particularly houses and offices. It provides virtualized scans of real indoor spaces like houses and offices. Using the Habitat platform, we place an agent at different locations and orientations in scenes from the Gibson dataset and store the views visible to the agent. We generate a set of images by obtaining several

views from the same scene. Therefore, together, each image set can be considered to represent the scene. Now, such scenes can be understood by asking and answering questions about the corresponding image sets.

From Gibson we collect two types of indoor scenes: 1.) Gibson-Building; and 2.) Gibson-Room. The first part of the dataset contains multiple images taken from the same building by placing the agent at random locations in a building and recording its viewpoint while the second (Gibson-Room) is collected by obtaining several views from the same room.

We show images from Gibson-Building sets to annotators and ask them to ask questions about the scene. We ask the annotators to try to ask questions which require at least two images to answer.

From a pilot study, we observed that it is easier for humans to frame questions if they are shown the full 3D view of a scene, simulating the situation of them being present in the scene and being able to move around. Humans are able to frame better questions about locations of objects, and their relationships when they are given complete information about a scene. Therefore, for Gibson-Room, we simulate such immersion by creating videos of the scenes by sequentially showing images from our image sets interspersed with intermediate view-points. This has the desired effect of providing a complete 360 degree view of the scene, albeit using many more images than required. We show these videos (see supplementary material for examples of how these videos are created) to the annotators and ask them to provide questions and answers about the scenes.

We obtain question-answer annotations for a scene by showing each such video

to several annotators using Amazon Mechanical Turk. We ask each annotator to ask a question about the scene and also provide the corresponding answer. We request that the annotators should ensure that their question can be answered using only the scene shown and no additional knowledge should be required.

Next, we randomly sample sets of frames from each video and associate the questions obtained from the video with these image sets. Unlike uniform sampling of the frames, with random sampling we cannot be sure that a question can be answered using an image set. Therefore, we refine the image-set data by obtaining annotations from other annotators as to whether a question can be answered using the image set provided, and if it can be, then what should be the answer. We discuss this step in Section 6.3.1.3.

6.3.1.2 Outdoor Scenes

We collect annotations for the nuScenes dataset similar to the Gibson-Building setting. We show the annotators images from an image set. These represent a 360 degree view of a scene. We, again, ask them to write questions and answers about the scene as before.

6.3.1.3 Refining Annotations

From the first step of data annotation, we noticed that even though it is likely that the questions can be answered using the randomly sampled frames from a video, we cannot be sure this is always true. Therefore, to ensure that we know if

a question can be answered or not, we showed all the image sets in our datasets and the associated questions obtained from the previous step to up to three other annotators. We asked them to provide an answer to the question based only on the image set shown. We also asked them to say "Not possible" if the question cannot be answered. This step has the added benefit of increasing the confidence about an answer if there is a consensus among the annotators. This is based on the idea of the wisdom of the crowd.

In addition, we also asked the annotators at this stage to mark the images which are required to answer the given question. This provides us another level of information about which images are the most salient for answering a question. Such information can potentially be used to guide models to select and focus more on the relevant images.

6.3.1.4 Train and Test Splits

After refining, we divided the datasets into training and testing splits. The statistics of these splits are given in Table 6.1. To create test splits, we select some samples for which at least two out of three annotators agreed on the answer. We also ensured that the train and test sets from the same dataset have the same answers (though distribution of answers might be different.)

Table 6.1: Statistics of the datasets. Each of our train and test sets is associated with six images, one question, and up to four answers.

Dataset	#Train sets	#Test sets	#Unique answers
Indoor - Gibson (room + building)	69,207	22,272	961
Outdoor - nuScenes	33,973	15,644	650

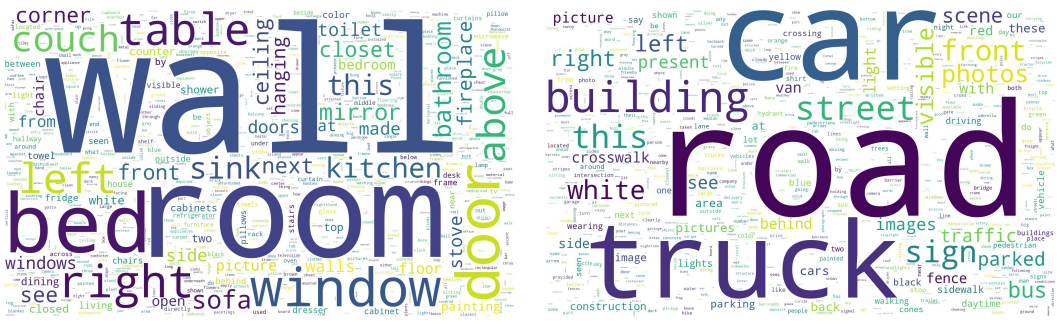


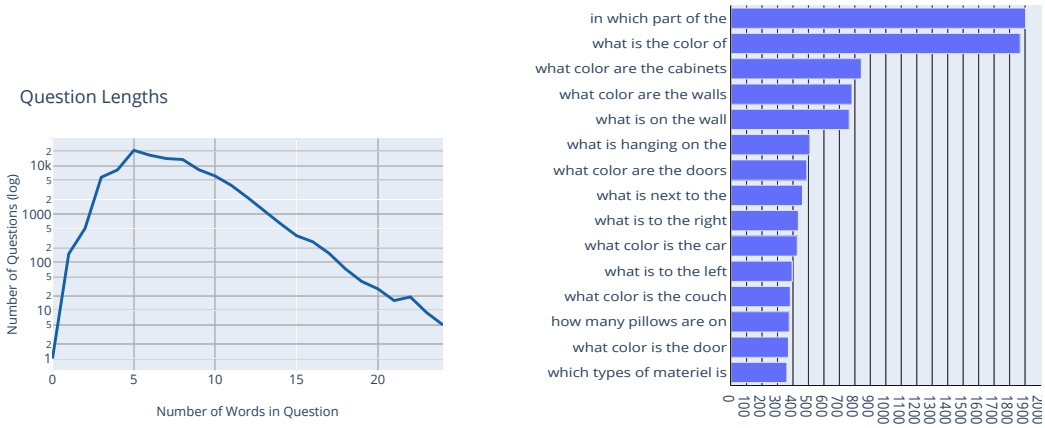
Figure 6.5: Question wordclouds for Gibson (left) and nuScenes (right) datasets. Note that these cover a large variety of concepts. We observe that for indoor scenes (left) people are interested in objects hanging on walls, kept on beds and couches, and furniture kept in rooms. For outdoor scenes, as dictated by intuition, we observe that most questions are about the road, cars, buses, and trucks.

6.3.2 Dataset Analysis

In this section, we present some salient features and statistics of our combined dataset which reveal unique properties of the datasets.

6.3.2.1 Question word distributions

The question word clouds for both parts of the dataset are shown in Figure 6.5. Note that we have removed the first few words from each question before plotting the question wordclouds. This gives us a better picture of which objects people are interested in. Clearly, for outdoor scenes, people are most interested in objects commonly found on the streets (cars, trucks, buildings) and their properties (types,



(a) Question lengths

(b) Most frequent questions

Figure 6.6: Statistics of the full dataset. (a) The distribution of questions over numbers of words. (b) The most frequent types of questions in the dataset.

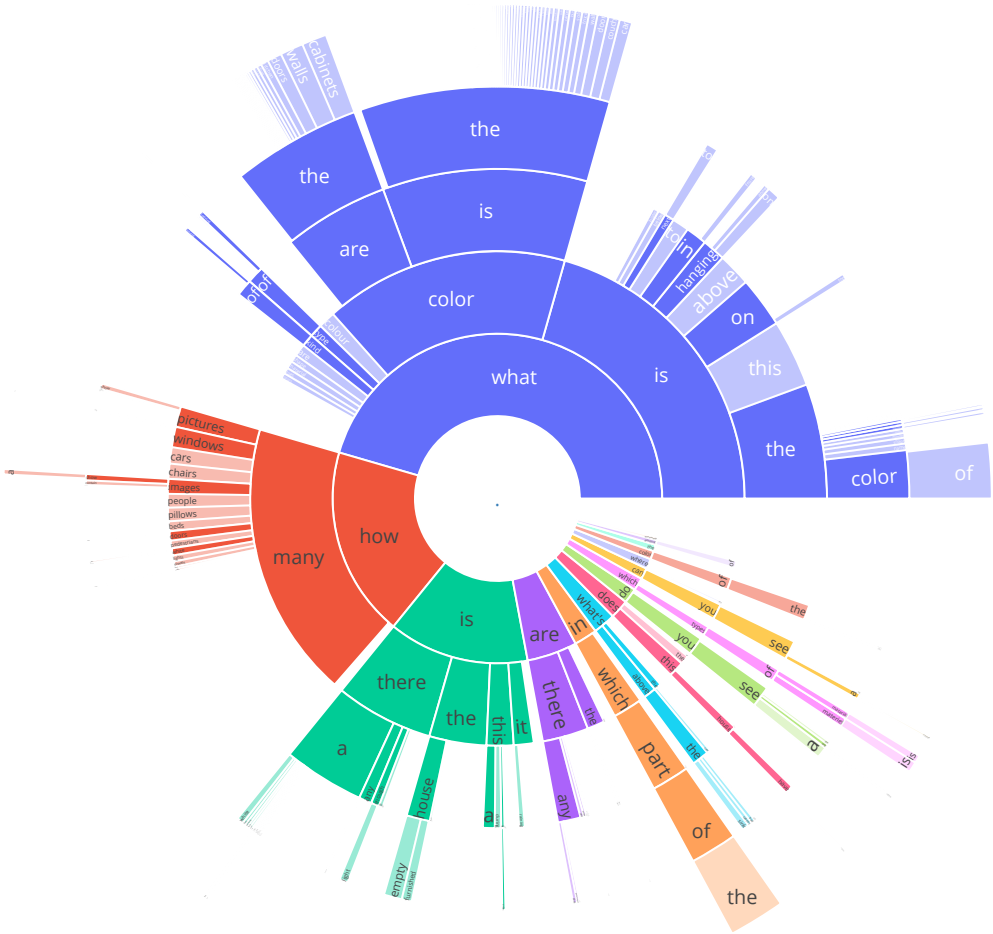


Figure 6.7: The most frequent first five words of the questions (from inside to outside). (Best viewed digitally)

colors, numbers). On the other hand, for indoor scenes, the most frequent questions are about objects handing on walls and kept on beds, and the room layouts in general.

6.3.2.2 Types of Questions

In Figure 6.6a, we plot the distributions of question lengths for the whole dataset (indoor scenes + outdoor scenes). We observe that a large chunk of the questions are between 5 and 10 words long. Further, in Figure 6.6b, we plot the numbers of the most frequent types of questions for the dataset. We observe that the most frequent questions are about locations of objects, properties of single objects, and spatial relationships between different entities.

To understand the types of questions in the dataset, we plot the distribution of the most frequent first five words of the questions in the whole dataset in Figure 6.7. Note that a large portion of the questions are about the numbers of different kinds of objects. Another major subset of the questions are about geometric relationships between objects in a scene. A third big part of the dataset contains questions about colors of objects in scenes. All of these are important types of questions. Answering questions about the colors of things in a scene requires localization of the object of interest. Depending on the question, this might require reasoning about the relationships between objects in different images. Similarly, counting the number of a particular type of object requires keeping track of previously counted objects to avoid double counting if the same object appears in different images.

Solving such questions simultaneously by reasoning across images demands developing new algorithms.

6.3.2.3 Answer Distributions

Figure 6.8 shows the distribution of answers in the dataset (combined Gibson and nuScenes) for frequently occurring questions types. Most types of questions do not have a dominant answer. Of particular note are the questions about relative locations and orientations of objects, e.g. "What is on the", "What is next to", and questions about the numbers of objects e.g. "How many cars are", "How many chairs are" etc. This means that it is difficult for a model to perform well by lazily exploiting the statistics of question types.

6.3.2.4 Number of Images Required

In the second stage of the annotation procedure, in addition to refining the annotations, we also collect annotations for which images are required for answering the given question. We ask the annotators to mark only those images in an image set which are required to answer the associated question and ignore the others. In Figure 6.9, we plot the histogram for the number of images required to answer each question for both Gibson-Room and Gibson-Building datasets. Note that, for the plot in Figure 6.9, we only consider those image sets for which at least 2 annotators agree about the images which are needed. Approximately one-third of the samples in the Gibson-Room dataset require at least two images to answer the question. As

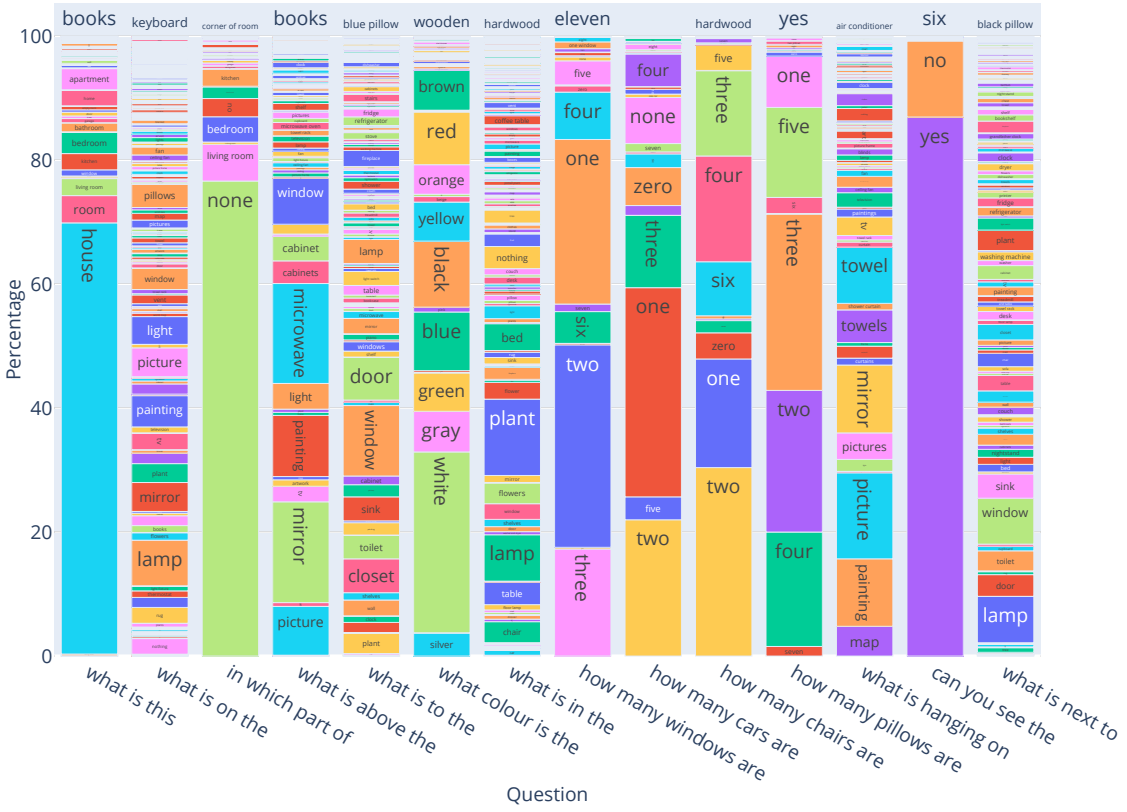
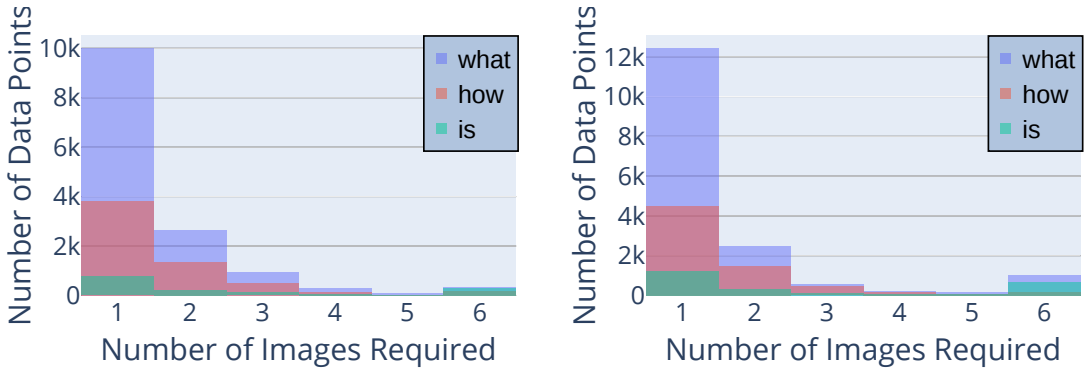


Figure 6.8: Answer distributions for several types of questions in the whole dataset. The questions plotted are among the most frequent. (Best viewed digitally)



(a) Gibson-Room (b) Gibson-Building

Figure 6.9: Histogram of the number of images required to answer different types of questions for a.) Gibson-Room; and b.) Gibson-Building. Observe that the Gibson-Building dataset is more skewed towards questions requiring only a single image to answer.

expected, this ratio is lower for Gibson-Building dataset. Many of the questions obtained require the answerer to find the single most-relevant image from the set and answer the question based only on that. It is difficult for humans to reason across images and relate multiple objects and view-points. We believe that this is especially true for hasty crowd annotators who are paid according to the number of tasks that they complete. This is still a challenging setting and involves rejecting most of the images in the image set and focusing only on one image. In theory, such questions can potentially be answered by using existing single-image VQA models.However, this would require the single-image VQA model to answer "Don't know" for all the irrelevant images and find only the most relevant one. Current VQA models do not have the ability to do this in most cases (see supplementary material for examples).

6.4 Experiments

6.4.1 Implementation Details

We use Detectron [68] to extract the region proposals and features R_i for each image. Each region feature is 2048-D and we use the top 100 region proposals from each image. To obtain the word-vector embeddings we use 300-D GloVe [158] vectors. The joint visual-question embedding, x is taken to be 5000-D. For evaluation, we use the VQA-Accuracy metric [14]. A predicted answer is given a score of one if it matches at least two out of the three annotations. If it matches only one annotations, it is given a score of 0.5. All of our models are implemented in the Pythia framework [191] and are trained on two NVIDIA V100 GPUs for 22,000 iterations with a batch size of 32. The initial learning rate is warmed up to 0.01 in the first 1,000 iterations. The learning rate is dropped by a factor of 10 at iterations 12,000 and 18,000.

6.4.2 Results

We report the answer accuracy for all baselines in Table 6.2. We observe that the the accuracy achieved by both of the VQA-based baselines is only around 50%. This highlights the need for advanced models for ISVQA.

Table 6.2: Baseline results for both datasets.

	Method	VQA-Accuracy (%)	
		Gibson	nuScenes
Prior-Based Baselines	Naïve	8.61	22.46
	Hasty-Student	27.22	41.65
	Question-Only	40.26	46.06
Baselines	Concatenate-Feature	47.57	53.66
	Stitched-Image	50.53	54.32

6.4.2.1 Comparison between Baselines

From Table 6.2, we observe that, as expected, the naïve baseline performs worst. It gives a VQA-Accuracy of only 8.6% for the indoor scenes (Gibson) dataset compared to 47.57% given by the Concatenate-Feature baseline and 50.53% given by the Stitched-Image baseline model. This shows that ISVQA presents unique challenges which cannot be overcome trivially.

6.4.2.2 Language Biases

Recent works (e.g. [8]) show that high performance in VQA could be achieved using only the language components. Deep networks can easily exploit biases in the datasets to find short-cuts for answering questions using only the language features. We observe that the VQA-based baselines perform far better than the question-only baseline. This shows that our datasets are not heavily biased and validates the utility of developing ISVQA models that can utilize both the visual and language components simultaneously.

6.4.2.3 Performance by Question Type

Figure 6.10 shows the accuracy bar-chart of our single-image VQA-based baselines for various types of questions. Using this chart, we have the following observations and hypotheses:

Single-image VQA baselines can predict single-object attributes. Both baseline models can answer questions about colors of single objects well (black and gray bars). This is expected because no cross-image dependency is needed.

General cases may need cross-image inference. A large portion of questions involve multiple objects, which may appear in different images. The two baselines using simple attention do not perform well on such questions. Neither of the baselines has a sophisticated mechanism to infer across images or do multi-step reasoning. The solution to ISVQA problems may need multi-step reasoning mechanisms that understand the geometry of the scene behind the images.

Stitched-Image captures cross-image dependency better. The Stitched-Image baseline allows direct pooling from regions on all images, which may result in sightly better ability to capture across-image dependency. It also outperforms the Concatenate-Feature baseline for most question types, except for the counting questions about objects likely to appear in multiple images. The Stitched-Image baseline does not have a mechanism to avoid double counting the same object.

This reveals the limitations of existing methods. VQA methods cannot reason about relative locations and orientations of objects across several images. VQA-

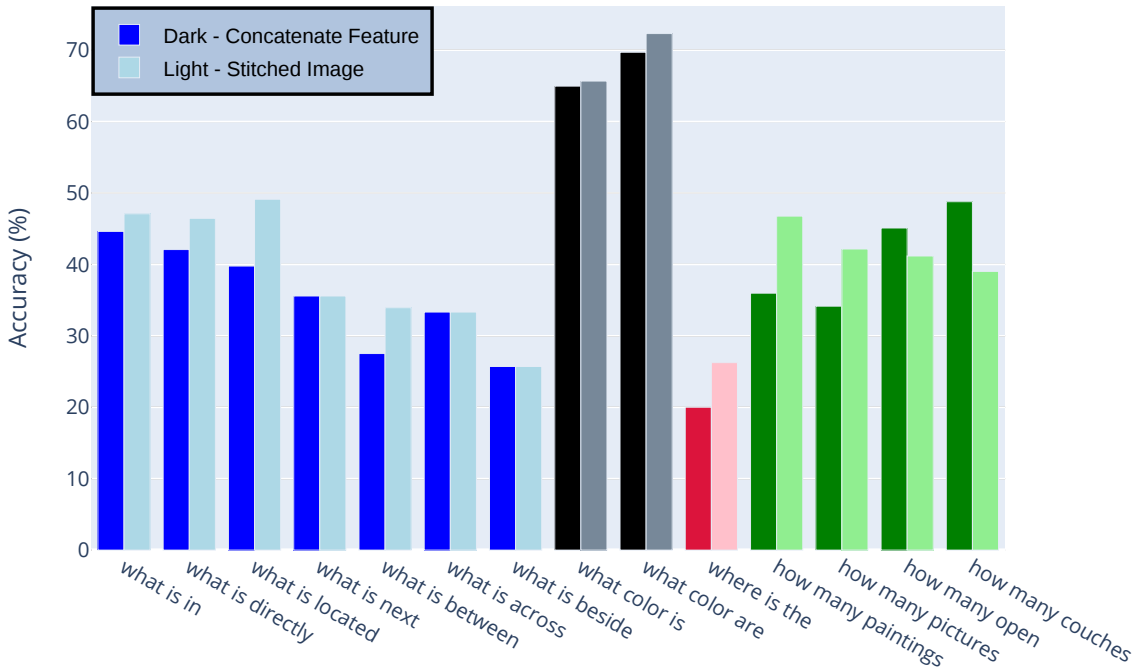


Figure 6.10: Performance of the two VQA-based baselines for different types of questions for the combined Gibson test set. Dark colors represent the performance for Concatenate-Feature baseline and light colors for Stitched Image baseline. **Blue** is used for geometric relationship questions, **green** for counting questions, **red** for location, and **black** for color questions. We notice that the VQA-based baselines are able to answer simple questions like those about colors of single objects very well. However, questions involving spatial reasoning between objects in one image or across images are extremely challenging for such methods.

based methods are not suited for such reasoning and methods specific to ISVQA need to be developed.

6.5 Discussion and Conclusion

We proposed the new task of image-set visual question answering (ISVQA). This problem can lead to new research problems like language-guided cross-image attentions and reasoning. To establish the ISVQA problem and enable its research, we introduced two ISVQA datasets for indoor and outdoor scenes. Large-scale annotations were collected for questions and answers with novel ways to present the scene to the annotators. We performed bias analysis of the datasets to set up performance lower bounds. We also extended a single-image VQA method to two simple attention-based baseline models. Their limited performance reveals the unique challenges of ISVQA, which cannot be solved trivially by the capabilities of existing models. Approaches for solving the ISVQA problem may need to pass information across images in a sophisticated way, understand the scene behind the image set, and attend the relevant images.

Chapter 7: Datasets and Decisions for Deep Face Recognition

Current deep convolutional neural networks are very high capacity representation models and contain millions of parameters. Deep convolutional networks are achieving state-of-the-art performance on many computer vision problems [84, 90, 119]. These models are data hungry and their success is being driven by the availability of large amounts of data for training and evaluation. The ImageNet dataset [177] was among the first large scale datasets for general object classification and since it's release has been expanded to include thousands of categories and millions of images. Similar datasets have been released for scene understanding [1, 238], semantic segmentation [47, 130], and object detection [47, 63, 177].

Recent progress in face detection, and recognition problems is also being driven by deep convolutional neural networks and large datasets [119]. However, the availability of the largest datasets and models is restricted to corporations like Facebook and Google. Recently, Facebook used a dataset of about 500 million images over 10 million identities for face identification [200]. They had earlier used about 4.4 million images over 4000 identities for training deep networks for face identification [199]. Google also used over 200 million images and 8 million identities for training a deep network with 140 million parameters [184]. But, these datasets are

not publicly available.

The academic community is at a disadvantage in advancing the state-of-the-art in facial recognition problems due to the unavailability of large high quality training datasets and benchmarks. Several groups have made significant contributions to overcome this problem by releasing large and diverse datasets. Sun et al. released the CelebFaces+ dataset containing a little over 200,000 images of about 10,000 identities [197]. In 2014 Dong et al. published the CASIA WebFace database for face recognition which has about 500,000 images of about 10,500 people [237]. Megaface 2 [151] is a recent large dataset which contains 672,057 identities with about 4.7 million images. YouTube Faces [217] is another dataset targeted towards face recognition research. It differs from other datasets in that it contains face annotations for videos and video frames, while other datasets contain only still images. In [156], the authors released a dataset of over 2.6 million faces covering about 2,600 identities. However, this dataset contains much more label noise compared to [197] and [237].

In addition to downloading and annotating face images from the internet, some work has also been done on synthesis of face images for augmenting existing datasets. In [141,143], the authors use 3D models of faces to modify existing images to generate novel poses and expressions. Similarly, [180] uses semantic segmentation to composite 3D face models on face images. These models can be manipulated to generate more training data. However, the advantage of these methods over collecting images from the internet has not been effectively demonstrated.

Despite the availability of these datasets, there is still a need for more publicly available datasets to push the state-of-the-art forward. The datasets need to be

Figure 7.1: Few samples from the dataset discussed in the chapter. Each column represents variations in pose and expression of images of a subject.

more diverse in terms of head pose, occlusion, and quality of images. Also, there is a need to compare performance improvements with deep data (fewer subjects and more images per subject) against wide data (more subjects but fewer images per subject).

In this chapter, we introduce two new large-scale datasets[1] which will facilitate the training of deep networks for face identification and verification.

7.1 UMDFaces Dataset

The first of these datasets has 367,888 face annotations of 8,277 subjects. Similar to [237], our dataset is wide and may be used separately or to complement the CASIA dataset. We describe the data collection and annotation procedures and compare the quality of the dataset with some other available datasets. We provide bounding box annotations which have been verified by humans. Figure 7.1 shows a

[1]Available from `https://www.umdfaces.io`

small sample of faces in the dataset for five subjects. We provide the locations of fiducial keypoints, pose (roll,pitch and yaw) and gender information generated by the model presented in [172]. In addition to this, we also provide human verified keypoint locations for 115,000 images.

We will discuss the second dataset later in the chapter.

7.1.1 Data Collection

Using the popular web-crawling tool, GoogleScraper [2], we search for each subject on several major search engines (Yahoo, Yandex, Google, Bing) and generate a list of URLs of images. We remove the duplicate URLs and download all the remaining images.

7.1.2 Face detection

We use the face detection model proposed by Ranjan et al. [170] to detect the faces in the downloaded images. Because we wanted a high recall, we set a low threshold on the detection score. We kept all the face box proposals above this threshold for the next stage.

7.1.3 Cleaning the detected face boxes by humans

Several bounding boxes obtained by the process discussed above do not contain any faces. Also, for each subject, there may be some detected face boxes which do not belong to that person. These cause noise in the dataset and need to be removed.

[2] `https://github.com/NikolaiT/GoogleScraper`

We used Amazon Mechanical Turk (AMT) which is a widely used crowd-sourcing platform to get human annotations. These annotations are then used to remove extraneous faces.

For each subject, we show six annotators batches of forty cropped face images. Out of these forty faces, thirty-five are face detections which we suspected were images of the target subject but are not sure and five are added by us and we know they are not of the target individual. We know the locations of these 5 'salt' images and use these to verify the quality of annotations by an annotator. We also display a reference image selected manually by us. The annotators were asked to mark all the faces which did not belong to the subject under consideration.

We evaluate the annotators by how often they mark the 'salt' images that were presented to them. For example, if an annotator did 100 rounds of annotations and of the 500 'salt' images presented he/she clicked on 496 of them, his/her vote was given a weight of 496/500.

To actually determine if a given image is of the target individual or not, we used the following heuristic which associated with every face a score between 0 and 1:

1. Obtain the three highest vote weights and respective votes of all the annotators that had to decide on this face and call them w_1, w_2 and w_3, and their respective yes (1) or no (0) votes v_1, v_2 and v_3. For example w_3 is the vote weight of the highest scored annotator for this face, who voted for v_3.

2. If $w_1 + w_2 > 0.8$, the final score of this face is $\sum_{i=1}^{3} w_i v_i / \sum_{i=1}^{3} w_i$

3. If $w_3 > 0.6$, make the final score of this face v_3.

4. Otherwise there is no reliable, robust answer for this face; try to annotate it again.

This score has the following interpretation: closer to 0 means there is a robust consensus that the image is of the target individual and closer to 1 means there is a robust consensus that it is an image not of the target individual.

After associating a score with every face we had, we selected faces whose score was lower than 0.3 (after considering the quality and quantity trade-offs) and removed all other faces from our dataset.

The procedure presented in this section allowed us to economically and accurately label all the faces we obtained.

In the next section we describe the method for generating other annotations.

7.1.4 Other annotations

After obtaining the clean, human verified face box annotations, we used the All-in-one CNN model [172] to obtain pose, keypoint locations, and gender annotations.

We give a brief overview of this model.

Figure 7.2 shows some examples of the annotations in our dataset generated by the All-in-one CNN.

To verify the performance of the keypoints generated by the above model, we show the generated annotations for 115,000 images to humans and ask them to mark the images with incorrect keypoint annotations. We show each face to two people

Figure 7.2: Some examples with annotations generated by the All-in-one CNN [172]. Blue box indicates that the estimated gender is male and a yellow box means that the estimated gender is female. Red dots are the detected keypoints and the green text is the estimated head pose (yaw, roll, pitch).

on Amazon Mechanical Turk (AMT). As a mark of the quality of the keypoints, we found that for about 28,084 images out of the 115,000 shown did both the annotators say that the keypoints are incorrectly located.

7.1.5 Final cleaning of the dataset

We notice that even after getting human annotations, the dataset still has some noisy face bounding boxes. For some individuals there are some boxes that belong to someone else or are not faces at all. Since we want to provide the cleanest dataset that we could, we remove these noisy boxes. Here we present the approach that we took to remove them.

We use the verification model proposed in [181] to remove the noise. For each subject , we extract the fc7 layer features and calculate the cosine distance between

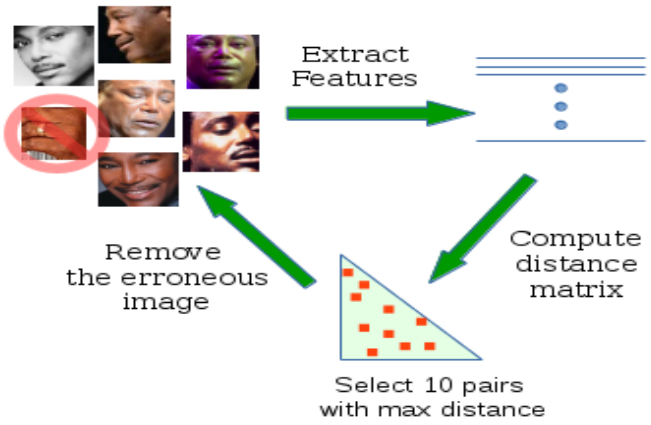


Figure 7.3: Overview of the strategy for final cleaning of the dataset. We adopt an iterative method (described in Section 7.1.5) for removing incorrect faces for each subject.

each pair of faces for that subject. We find the ten pairs with the maximum distance between them and sum these ten distances. We observe that if this sum is below a certain threshold (ten in our tests), then all the pairs are actually images of the same person. However, if the sum is above the threshold, then most of the times there is at least one noisy face box in the data for that subject. So, if the sum of distances is above the threshold, we find the face image that occurs in the maximum number of pairs out of the ten pairs selected and remove that image from the dataset. If more than one image occurred the maximum number of times, then we remove the one which contributes the most to the sum. We again calculate the similarity matrix and repeat the process till the sum of the ten pairs goes below the threshold. Figure 7.3 summarizes this approach.

If the above procedure leads to the removal of more than five images for a subject then we remove that subject identity completely. Using this process we removed 12,789 images and 156 subject identities from the dataset. Finally, our

dataset has 367,888 face annotations spread over 8,277 subject identities.

The process of training a face recognition system starts with choosing a dataset of face images, detecting faces in images, cropping and aligning these faces, and then training deep networks on the cropped and possibly aligned faces. Every step of the process involves many design issues and choices.

Some issues have received significant attention from researchers. These include choices about the architecture of neural networks. On the other hand, there are several other design choices that require more attention. These arise at every stage of the process from face detection and thumbnail (image obtained after cropping and aligning the face image) generation to selecting the training dataset itself. We study some of these design questions. For this, we now introduce a dataset of 22,075 videos collected from YouTube of 3,107 subjects. These subjects are mainly from batch-1 of the UMDFaces [18] dataset discussed above. We release face annotations for 3,735,476 frames from these videos and the corresponding frames separately. We use this dataset to study the effect of using a mixture of video frames and still images on verification performance for unconstrained face datasets such as the IJB-A [107] and YTF [217] sets.

Face detection is the first step in any face recognition pipeline. Several CNN-based face detectors have been introduced which achieve good detection performance and speeds [89, 170–172, 250]. Each of these detectors learns a different representation. This leads to generation of different types of bounding boxes for faces. Verification accuracy can be affected by the type of bounding box used. In addition, most

recent face recognition and verification methods [34, 87, 181, 195, 196, 199] use some kind of 2D or 3D alignment procedure [82, 104, 172, 252]. All these variables can lead to changes in performance of deep networks. To the best of our knowledge there has been very little systematic study of effects of the thumbnail generation process [156] on the accuracy of deep networks. In Section 7.3.4 we study the consequences of using different thumbnail generation methods. We show that using a good keypoint detection method and aligning faces both during training and testing leads to the best performance.

Other questions concern the dataset collection and cleaning process itself. The size of available face datasets can range from a few hundred thousand images [18, 152, 217, 237] to a few million [72, 73, 105, 151, 156]. Other datasets, which are not publicly available, can go from several million faces [199] to several hundred million [184]. Much of the work in face recognition research might behave differently with such large datasets. Apart from datasets geared towards training deep networks, some datasets focus on the evaluation of the trained models [91, 107]. All of these datasets were collected using different methodologies and techniques. For example, [217] contains videos collected from the internet which look quite different from still image datasets like [18, 73, 237]. We study the effects of this difference between still images and frames extracted from videos in Section 7.3.1 using our new dataset. We found that mixing both still images and the large number of video frames during training performs better than using just still images or video frames for testing on any of the test datasets [18, 107, 217].

In Section 7.3.2, we investigate the impact of using a deep dataset against

using a wider dataset. For two datasets with the same number of images, we call one deeper than the other if on average it has more images per subject than the other. We show that the choice of the dataset depends on the kind of network being trained. Deeper networks perform well with deeper datasets and shallower networks work well with wider datasets.

Label noise is the phenomenon of assigning an incorrect label to some images. Label noise is an inherent part of the data collection process. Some authors intentionally leave in some label noise [72, 73, 156] in the dataset in the hopes of making the deep networks more robust. In Section 7.3.3 we examine the effect of this label noise on the performance of deep networks for verification trained on these datasets and demonstrate that clean datasets almost always lead to a significantly better performance than noisy datasets.

7.2 UMDFaces-Videos Dataset

Still photos from the internet cannot match the amount of variation that videos provide. Videos (and frames extracted from the videos) are under-utilized because of the difficulty in cleaning and annotating the data. There is a need for effective methods for annotating video data. We describe a new dataset aimed at face recognition research. It contains 22,075 videos of 3,107 subjects collected from YouTube. We provide bounding box annotations for 3,735,476 frames from the videos. We explain our methodology of collecting this dataset which, we hope, will be useful to researchers working on face verification and related problems.

Figure 7.4: Some sample annotated bounding boxes from the dataset. Each row contains frames from a video. There is a large amount of pose and expression variation in each video.

7.2.1 Data Collection

We search YouTube for over 3000 subject identities (from batch-1 of UMDFaces [18]) and try to download the first 20 videos for each person. We use the open source system youtube-dl [4] for searching and downloading the videos. We downloaded a total of about 40,000 videos.

7.2.2 Automated filtering

From each video, we extract either all the frames or the first 4,000 frames, whichever is lower. This process gave us over *140 million* frames. We randomly select about 10% of these frames to process further. Next we detect faces in the retained frames. We use the YOLO detector [173] for detecting faces. We train YOLO on the WIDER

dataset [231] and fine-tune on FDDB [95]. This gives us over 40 million face boxes in 14 million frames. We again randomly select 4000 face boxes for each subject identity finally leaving ourselves with about 14 million boxes.

Our next task is to remove all face box proposals which do not belong to the person in question. We again use the All-in-one method [172] to detect key landmark points on each face and use them to align the faces. We use the images in batch-1 of the UMDFaces dataset [18] as reference images for the subjects. Our problem now reduces to a verification problem. For each subject we need to verify whether a face box belongs to that person.

We use the verification method proposed in [34] for filtering the proposal boxes which are not of the person in question. We extract features (using a network trained in the same way as [34, 237]) for all images in batch-1 of UMDFaces [18] and take their average over a subject to obtain one feature vector for the subject. Then, for each face box in our dataset for the subject, we compare the feature vector with the reference feature vector obtained above and keep only those boxes with similarity with the reference feature vector above a threshold. We use cosine similarity as the similarity metric and use a low threshold to avoid removing the hard-positive examples from the dataset as these are valuable. This leaves us with about 4 million face boxes.

7.2.3 Crowd-sourcing final filtering

To obtain the final dataset, we use Amazon Mechanical Turk (AMT) to filter the proposals. We show each proposal to 2 'mechanical turkers'. Each screen in our AMT task contains 50 images to be filtered and 3 reference images of a subject obtained from the UMDFaces dataset [18]. We requested the mechanical turkers to select images which do not belong to the subject under consideration. We remove all the faces boxes which are selected by at least one turker. To ensure high quality annotations, we adopt the following quality control method.

7.2.4 Quality control through sentinels

We used the method similar to the one used earlier for controlling the quality of annotations. Each screen of 50 images contains 5 known images of another subject. Depending on whether the turkers select these 'sentinel' images, they get an accuracy score. We only considered the votes of turkers with high accuracy scores.

After the final filtering through human annotators, we have 3,735,476 annotated frames in 22,075 videos.

7.3 Questions and Experiments

We show that judicious decisions about the training set and procedures can lead to large improvements in verification accuracy of deep networks. We first use the introduced dataset to show the importance of using video frames while training for verification. Then we investigate some more questions that will guide researchers

towards good practices for training deep networks for face verification and identification. These include: (i) whether deep datasets are better than wide datasets (Section 7.3.2); (ii) whether label noise helps in improving performance (Section 7.3.3); and (iii) how important is the thumbnail generation method for training and testing deep networks (Section 7.3.4).

7.3.1 Do deep recognition networks trained on stills perform well on videos?

Images in most still image datasets [91,114,136] are taken with high quality cameras in good lighting. Photos of celebrities on the internet are often selected from among several taken by a professional photographer. This introduces a bias towards high quality images. Models trained on only still images perform poorly on frames extracted from videos [107]. These frames are challenging because of pose, expression, and lighting variations. At the same time, models trained only on videos perform poorly on still images. There is a huge amount of video data available and only a limited number of still images. We show that training on a mixture of images and video frames is really important for achieving good verification performance.

We train deep networks on the following five sets and compare the verification performance of these networks:

- **Stills**: Some part (batch-1) of the UMDFaces [18] dataset. This comprises of about 140,000 still images. We train an Alexnet-derived architecture [181] on these images for 100,000 iterations with a batch size of 128 and initial learning

rate of 0.01 and reduced by half every 15,000 iterations.

- **Frames**: The same number (140,000) of video frames from our dataset (Section 7.2). Each subject has the same number of images as the case above (Stills). We used the same training method as above.
- **Frames++**: The same number of subjects as above but using many more video frames per subject for a total of about 1 million video frames. We trained this model for 100,000 iterations and decreased the learning rate by half every 20,000 iterations.
- **Mixture**: A mixture of still images and video frames from UMDFaces and our dataset. We took 50% of images from batch-1 of UMDFaces and the other 50% from our video frames dataset for a total of about 140,000 images. We trained this network for 100,000 iterations.
- **Mixture++**: The same number of still images as 'Stills' but about 1 million video frames. We again train the network on this dataset for 100,000 iterations.

Note that we are using far more images in the Frames++ and Mixture++ cases than the other cases. However, we believe that it is fair to compare these five methods because it is much easier to obtain millions of video frames than to obtain millions of still images. There is a lot more variation in 100 images than there is in 100 continuous video frames. Also, in real world scenarios, the amount of video data is increasing rapidly and the majority of recognition has to happen in videos.

We use an architecture [181] derived from Alexnet [113] due to it's easy availability and practicality. It is fast to train and is perfectly suited for large-scale

experiments like ours. However, we believe that, in this case, our observations are general and will be valid for other network architectures too.

We train networks on these five sets and compare performance of the trained models on IJB-A [107], YouTube Faces datasets [217] and batch-3 of the UMDFaces [18] dataset. In this experiment and the rest of the chapter, unless otherwise stated, we train the same architecture of networks on different datasets for a fixed number of iterations (100,000). We adopt the 1:1 verification protocol similar to the one introduced in [107] for evaluating the performance of these deep networks. We give a brief description of the evaluation protocol next.

7.3.1.1 Protocol

The IJB-A 1:1 verification protocol [107] uses a decision error trade-off (DET) curve for evaluation. The DET curve is equivalent to an ROC curve. In our examples we evaluate the performance for 1:1 verification on pairs of images or templates for different datasets [18,91,217]. For all experiments, we use ROC curves for evaluation.

7.3.1.2 Results

Figures 7.5 and 7.6 show the performance of the above five experiments. They clearly show the importance of using a mixture of video frames and still images for all cases. We see that while the performance of the 'Stills' and 'Mixture' cases is close for both IJB-A and UMDFaces, the performance of 'Frames' is poor. This is because of the presence of many still images in the test sets and the low variety

in the few training frames. On the other hand, note that the performance of the 'Mixture++' case is much better than any other case, even better than 'Frames++' which has similar number of images. This shows the importance of using both still images and the ample number of frames extracted from videos for improving verification performance on unconstrained faces.

Also, note from Figure 7.6 that when testing on a dataset which contains a mixture of still images and video frames [107], the performance of 'Mixture++' is the highest and 'Frames++' is the second highest. However, when testing on the UMDFaces dataset [18] which contains only images, 'Stills' performs second best after 'Mixture++' (Figure 7.5). Similarly, when testing on the completely video-based testing set YTF [217], from Figure 7.7, 'Mixture++' performs the best and 'Frames++' performs a bit worse than it. Also note that 'Mixture' performs better than 'Stills' and 'Frames'. Collecting millions of still images with enough variations is extremely difficult. It is much easier to collect and annotate millions of video frames. Also, using a combination of a large number of video frames and relatively few still images gives a significant boost in performance over using only still images or video frames.

7.3.2 What is better: deeper or wider datasets?

For datasets with the same number of total (still) images, we call a dataset with more images per subject deeper than another dataset with fewer images per subject. We call the latter dataset wider than the prior. An example of a deep (deeper than many

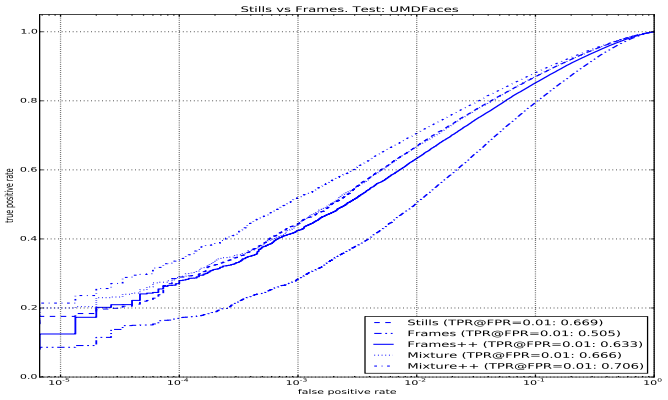


Figure 7.5: Verification performance of networks trained on 'Stills', 'Frames', 'Mixture', 'Frames++', 'Mixture++' and tested on UMDFaces batch-3 [18]. Note that the test set comprises of only still images. The performance of 'Stills' and 'Mixture' is similar. However, 'Mixture++' performs best. 'Stills' performs the next best after 'Mixture++' in this case.

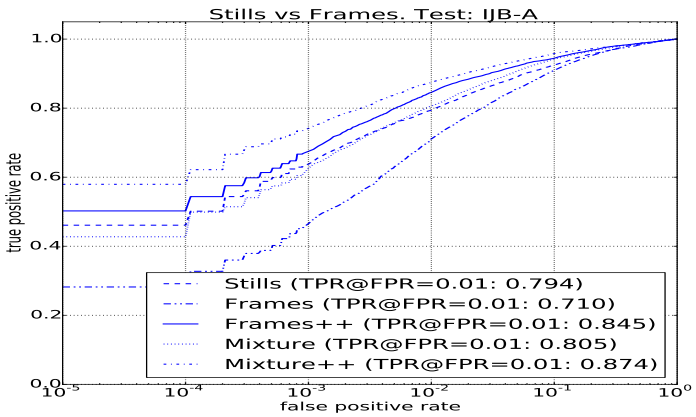


Figure 7.6: Verification performance of the five networks (Stills, Mixture, Frames, Mixture++, and Frames++) on IJB-A test set [107]. The IJB-A test set contains a mixture of still images and video frames. Again, the performance of 'Stills' and 'Mixed' are almost the same and 'Mixture++' is better than everything else. However, unlike Figure 7.5, the performance of 'Frames++' is better than 'Stills'.

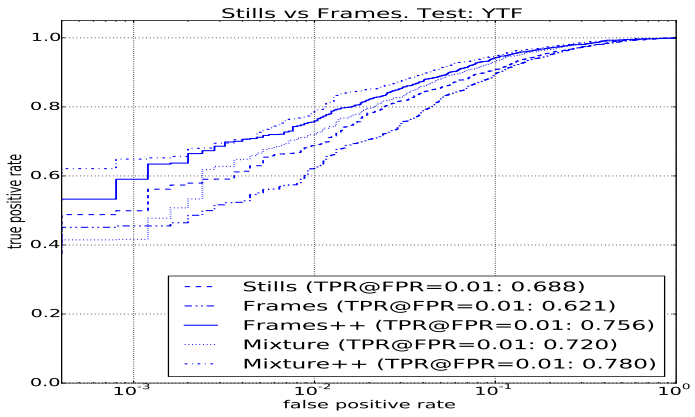


Figure 7.7: Verification performance of the five networks (Stills, Mixture, Frames, Mixture++, and Frames++) on YTF test set [217]. The test set contains only frames extracted from videos. Again, the performance of 'Mixture++' is better than everything else. Also, 'Mixture' performs better than 'Stills' in this case.

other still image datasets) dataset is the VGG-Face dataset [156] which has about 2.6 million images of 2,622 subjects. On the other hand CASIA-WebFace [237] can be considered a wide dataset. An extreme example of a wide dataset is the MegaFace training dataset [105, 151] which has over 670,000 subjects and only about 7 images per subject.

It is not intuitively clear whether it's better to use deeper datasets or wider for training deep networks. Given enough images, both deep and wide datasets can contain a variety of face images. Deep datasets are more varied in pose, expression, illuminations etc. On the other hand wide datasets contain large variations because of the large number of unique identities. In this section, we try to resolve the dilemma of choosing one kind of dataset over the other.

We use the UMDFaces [18], MS-Celeb-1M [73] and CASIA-WebFace [237] datasets to analyze the question. We treat batch-1 and batch-2 of UMDFaces as

the training set. To explore the question of deeper vs wider datasets, we divide the training datasets into two as follows: We sort the subjects according to the number of images they have; then we start with the subject with the maximum number of images and put the subject in one set (head); we then take the subject with next highest number of images and add him/her to the head set; we continue this process till we have collected close to half the total number of images. Now we have divided each dataset into two parts. The first part (which we call 'head') contains the deeper half of the dataset. The other half is called the 'tail'. For CASIA-WebFace, the 'head' dataset contains 1,738 subjects and 247,196 images and the 'tail' set contains 8,437 subjects and 247,218 images. Similarly, the UMDFaces 'head' set has 2,142 subjects with 144,371 images and the 'tail' set has 4,092 subjects and 144,348 images.

We first train the same architecture networks on the 'head' and 'tail' sets of both CASIA-WebFace and UMDFaces. We test these networks using the protocol from Section 7.3.1.1 on the UMDFaces batch-3 [18], IJB-A [107], and Labeled Faces in the Wild (LFW) [91] datasets. The results are shown in figures 7.8, 7.9, and 7.10. We note that the performance of the network trained on the 'tail' sets is better than the corresponding network trained on the 'head' sets for all three test sets. This means that, for a given number of images, it is better to have more subjects than having more images for fewer subjects.

On the other hand, if we train deeper networks, the performance of networks trained on the 'head' sets is better than the corresponding network trained on the 'tail' sets. This can be seen in Figure 7.11 where we train ResNet-101 [84] networks

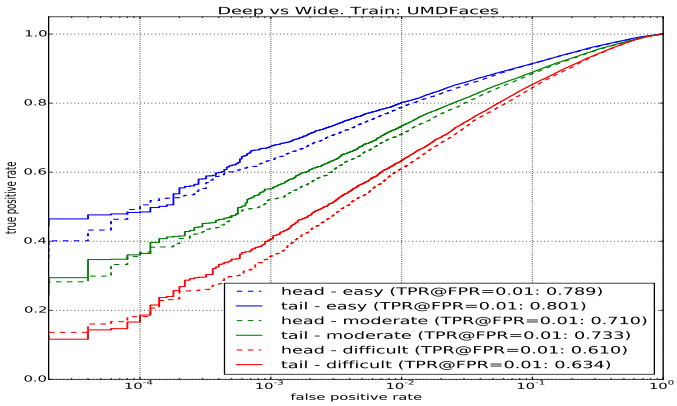


Figure 7.8: Training on UMDFaces [18] batch-1 and batch-2 and testing on batch-3. Solid lines represent training on the 'tail' (wide) set and dashed lines represent training on the 'head' set. We show the performance over three parts of the test dataset: easy, moderate, and hard. These parts are based on the difference in pose of the pair of images. The performance of the network trained on the 'tail' set is invariably better.

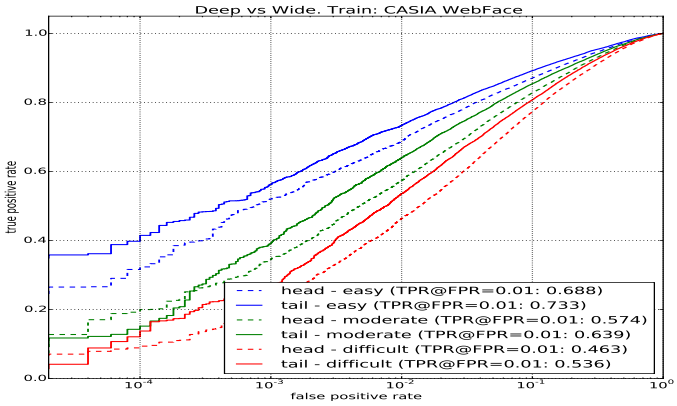
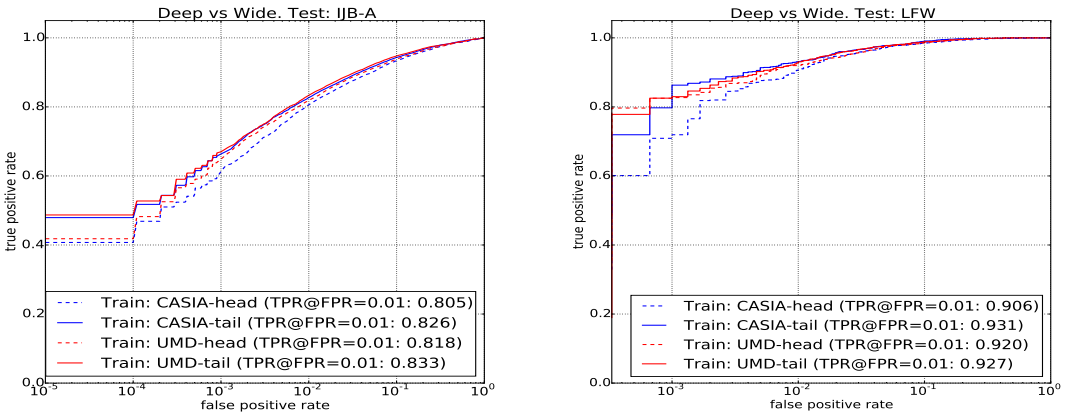


Figure 7.9: Verification performance of the networks trained on CASIA-WebFace [237] 'head' and 'tail' sets. We see similar trends as Figure 7.8.



(a) IJB-A (b) LFW

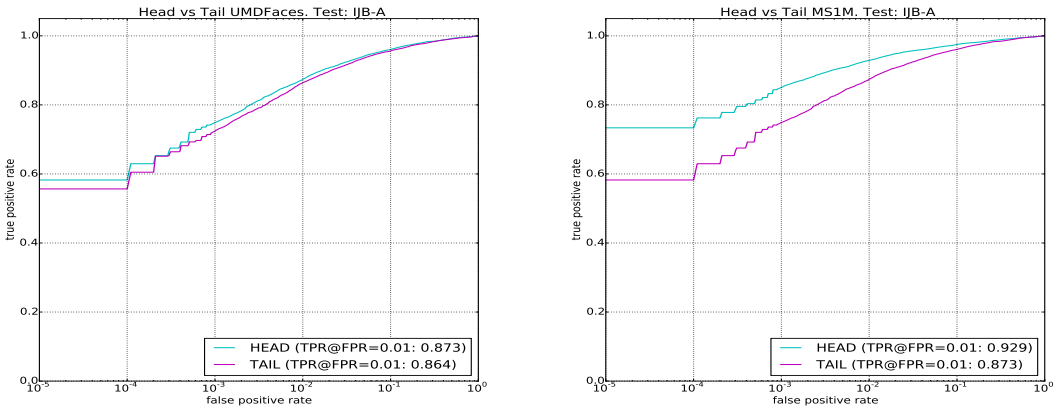
Figure 7.10: Performance on (a) IJB-A [107] and (b) LFW [91] of the networks trained on CASIA [237], and UMDFaces [18] 'head' and 'trail' sets. The performance of the networks trained on 'tail' are better across the range of false positive rate. (Best viewed digitally)

on the 'head' and 'tail' sets of UMDFaces [18] and MS-Celeb-1M [73] datasets and test on the IJB-A protocol [107].

This observation is important because it can guide researchers towards better practices to follow while collecting data or selecting data for training deep networks. Data acquisition is an expensive and time consuming process and these experiments shine a light on how to obtain the maximum benefit from the investment. This is an interesting direction for future work.

7.3.3 Does some amount of label noise help improve the performance of deep recognition networks?

In face identification and verification research, the effect of label noise in the training set for deep networks has not been studied extensively [72,73]. Label noise essentially means that some of the images have incorrect labels. Some [73,156] have suggested



(a) UMDFaces (b) MS1M

Figure 7.11: Performance on IJB-A [107] of ResNets trained on UMDFaces [18], and MS-Celeb-1M [73] 'head' and 'trail' sets. The 'head' sets are better.

that deep networks are robust to label noise.

We again use CASIA-WebFace [237] and UMDFaces [18] batch-1 and batch-2 for training the networks and LFW [91], IJB-A [107], UMDFaces batch-3 for evaluating the performance of these trained networks. We use the protocol explained in Section 7.3.1.1 for evaluation.

For both training datasets, we train recognition networks with $0, 2\%, 5\%$, and 10% label noise in the dataset. We would like to point out these percentages assume that the original datasets do not already contain any label noise. This assumption might not be true for many face datasets like MS-Celeb [73] and VGG-Face [156] which already contain some label noise.

Figure 7.13 shows the verification performance of networks trained on the CASIA-WebFace dataset for the UMDFaces test set and Figure 7.12 shows the same for networks trained on UMDFaces dataset. There is a clear degradation in performance with increasing noise level. For both datasets, the performance of the

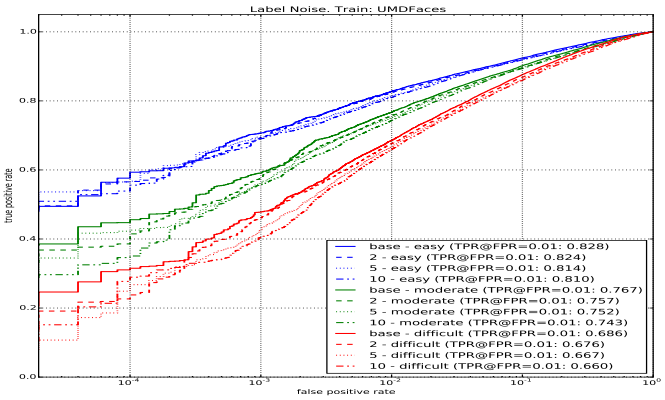


Figure 7.12: Verification performance on UMDFaces [18] batch-3 of deep networks trained on batch-1 and batch-2 with different noise levels. The colors represent the difficulty of test set (in terms of the difference in pose). Different line types represent different amounts of label noise added to the train set. Except for a small region in easy cases, using clean data is better than using data with label noise.

network trained on clean data is mostly better than the performance of networks trained with even small amounts of noise. Label noise does not improve performance over clean data for face recognition. However, the difference in performance between networks trained on clean data and data with low levels of label noise is relatively low. But the percentage of noisy labels should be relatively low (less than 5%) because from figures 7.13 and 7.12, we notice that for a label noise level of 10%, the performance invariably declines.

Similar trends can be seen for the LFW dataset in Figure 7.14b. However, when testing on the IJB-A dataset [107], we notice that this observation does not hold, as shown in Figure 7.14a. We believe that this is because the IJB-A protocol comprises of video frames which introduces another dimension of complexity for the model. Sometimes these video frames might not look like the person under consideration. We believe that such frames might be acting like a kind of label noise

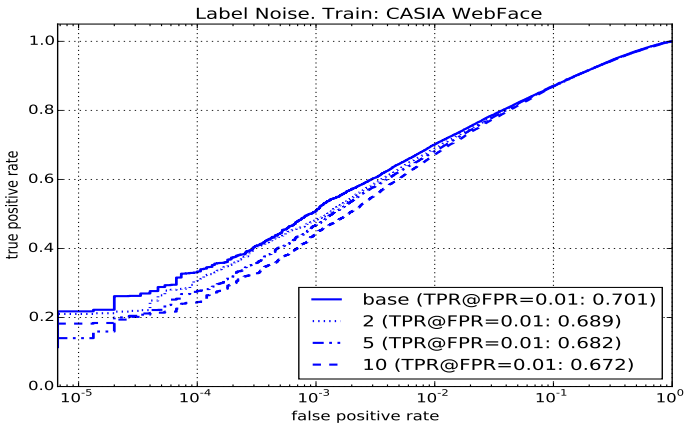
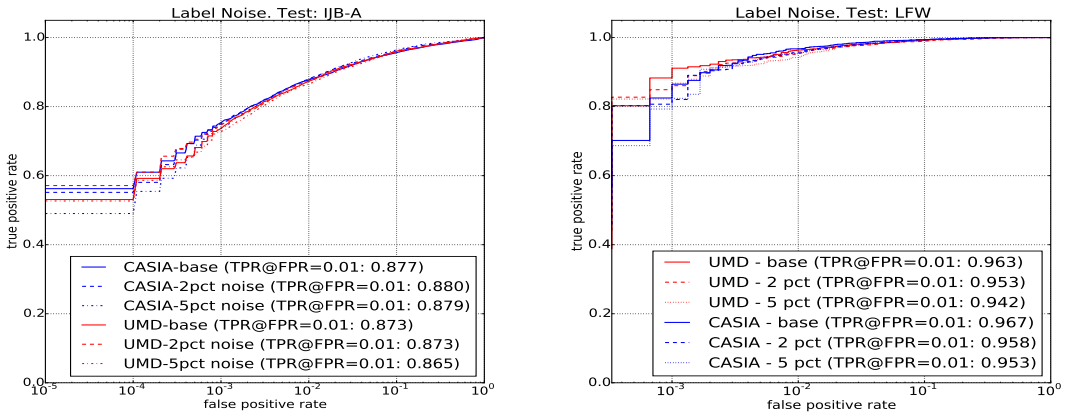


Figure 7.13: Performance on UMDFaces batch-3 for networks trained on CASIA-WebFace [237]. Similar to Figure 7.12 the network trained with no label noise performs best.



(a) UMDFaces (b) CASIA-WebFace

Figure 7.14: Verification results on IJB-A [107] of networks trained on (a) UMDFaces [18], and (b) CASIA WebFace [237]. Contrary to earlier observations from figures 7.12 and 7.13, the performance on IJB-A seems to improve with adding some label noise to the train dataset. (Best viewed digitally)

in the test set. That is why adding label noise to the training set might make the networks robust to such frames. However, label noise and its removal are definitely problems worthy of further research.

7.3.4 Does thumbnail creation method affect performance?

Detecting [33, 89, 95, 171, 207, 231], cropping, and aligning the faces in the dataset is the first step in many face recognition pipelines. Alignment is the process of transforming a face into some canonical view. This is usually done by detecting locations of keypoints [104, 172] in the face image and then using some kind of similarity transform to transform the faces to a canonical view [156]. We refer to the images of faces obtained after cropping and/or alignment as 'thumbnails'.

We investigate whether the performance of deep recognition networks is affected by the thumbnail generation process. We compare two popular alignment techniques [104, 172] against simple thumbnail generation techniques which only require keypoint locations and do not involve calculating any similarity transforms.

We compare three different types of thumbnails for evaluating verification performance. These are: (i) Keypoints from All-in-one CNN [172] with similarity transform alignment, (ii) DLIB keypoint detection and alignment [104], and (iii) Bounding box using keypoints from [172] without any alignment. In each case, we also study the effect of using tight thumbnails (tight crop of the face) vs loose thumbnails (including more context information). We try these methods for both training and testing and present the accuracies for the best performing cases in

figures 7.15 and 7.16. We use two different datasets for training: batch-1 and batch-2 of UMDFaces [18] and CASIA-WebFace [237], and UMDFaces batch-3 for evaluating the performance of the trained networks.

All the above mentioned variations give us the following seven methods of thumbnail generation: (1) loose alignment using [172] keypoints (*aligned_uf_loose*), (2) tight alignment using [172] keypoints (*aligned_uf_tight*), (3) loose alignment using [104] keypoints (*aligned_dlib_loose*), (4) tight alignment using [104] keypoints (*aligned_dlib_tight*), (5) no alignment with extremely tight crops (max extent of the keypoints minus 10% of the height and width from both sides) based on keypoints obtained from [172] (*unaligned_uf_minus_10*), (6) no alignment with moderately tight crops (max extent of the keypoints) based on keypoints obtained from [172] (*unaligned_uf_tight*), and (7) no alignment but loose crops of the faces (max extent of the keypoints plus 10% of the height and width on both sides) using keypoints from [172] (*unaligned_uf_plus_10*).

We train neural networks on UMDFaces [18] and CASIA-WebFace [237] using these 7 thumbnail generation methods and test on batch-3 of UMDFaces [18] using the same 7 different thumbnail generation methods. Hence, for both training sets, we have 49 (7×7) pairs of train and test sets. For both training sets, we select the seven pairs which give the highest performance and plot them in figures 7.15 and 7.16. We note that there is a clear dependence of performance on the type of thumbnail used for training and testing. Using a good keypoint detection method and alignment procedure for both training and testing is essential for achieving good performance. Note that using a tight alignment using keypoints detected using [172] for both

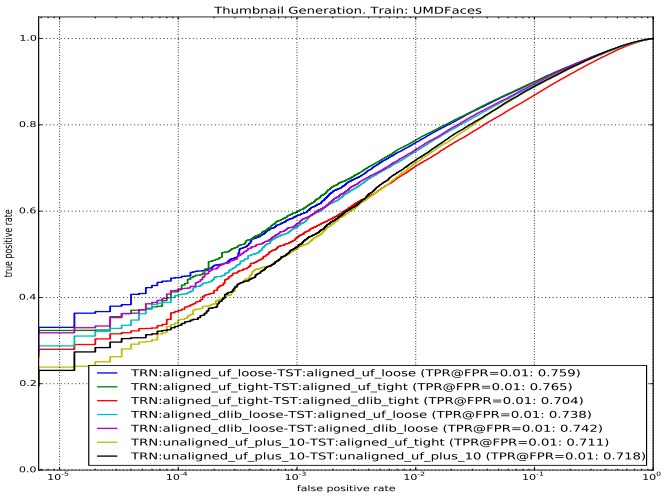


Figure 7.15: The performance of seven sets of train and test thumbnail generation methods. These seven were selected among all pairs of train-test pairs possible as explained in Section 7.3.4. The training set was the UMDFaces [18] dataset in each case and the testing set was batch-3 of UMDFaces. It is clear that tightly aligning both training and testing sets using the method from [172] gives the best performance (green). (Best viewed digitally)

training and testing gives the best performance among all the cases of networks trained on UMDFaces. This pair is also a close second among networks trained on CASIA-WebFace. As keypoint detection and alignment methods continue to improve, we expect the face verification performance to improve too.

7.4 Discussion and Conclusion

In this chapter, we studied the effects of certain decisions about datasets and the training procedures for training deep convolutional neural networks for face verification. Carefully making these decisions is important for developing face recognition systems. This work provides some guidelines about the decision making process.

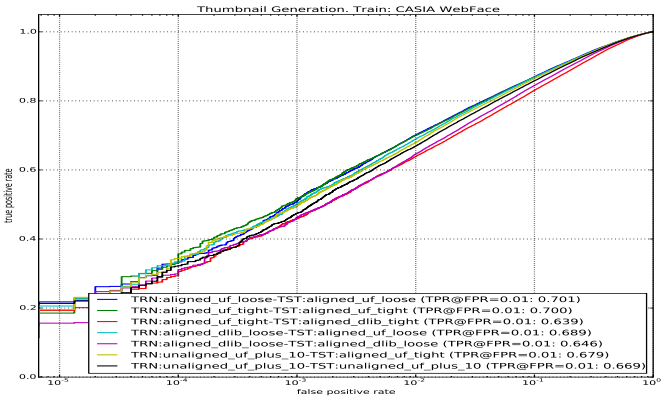


Figure 7.16: The performance of seven sets of train and test thumbnail generation methods with CASIA-WebFace as the training set and UMDFaces batch-3 [18] as the test set. We again see that aligning both training and testing sets using [172] gives the best performance. Also, using a loose alignment gives the best performance (blue) just slightly ahead of using a tight alignment (green).

There is an abundance of video data which contain much more pose and expression variations than still images. To ensure that researchers can take advantage of this potential, we introduced two new datasets: UMDFaces and UMDFaces-Videos. The UMDFaces-Videos datasets consists of of 22,075 videos and 3,735,476 annotated frames. The importance of removing label noise from the dataset and selecting wider or deeper datasets cannot be ignored. Similarly, aligning faces using accurate keypoints during both training and testing gives a boost in performance.

In the next chapter, we use the datasets introduced in this chapter and the insights gained from our experiments to develop fast and accurate face recognition systems. We will show that careful design of such face recognition systems can lead to significant performance gains over existing state-of-the-art.

Chapter 8: Learning Face Representations for Face Verification

In this chapter, we describe a complete algorithm designed using our UMDFaces datasets and the insights gained from the analysis above. We show that our algorithm can achieve state-of-the-art results for most recent benchmarks. We start by briefly describing face verification pipeline and the dataset used for training our model. We then discuss the loss function used. Finally, we present results for a variety of challenging, large-scale face verification and identification benchmarks and compare the results with existing state-of-the-art models.

8.1 Face Verification Pipeline

Given an image, we first detect all the faces using the DPSSD face detector [168]. Then, we crop out all the detected faces from the image and pass them through the All-In-One face [172] network to extract facial key-points. These key-points are used to align the corresponding faces in canonical coordinates. The aligned faces are then passed through our face DCNN, trained using Crystal Loss [168], to generate feature descriptors which are later used for verifying or identifying a face.

The proposed system for face identification and verification uses the All-in-One Face framework [172] for keypoint localization. The All-In-One Face is a recent

method that simultaneously performs the tasks of face detection, landmarks localization, head-pose estimations, smile and gender classification, age estimation and face recognition and verification. The model is trained jointly for all these tasks in a multitask learning framework, which builds up a synergy that helps in improving the performance of individual tasks.

Due to the lack of a single dataset which contains annotations for each task, various sub-networks are trained with different datasets. These sub-networks share parameters among them. This ensures that the shared parameters adapt to all the tasks instead of being task-specific. These sub-networks are fused into a single All-in-One Face CNN at test time. The complete network is trained end-to-end using task-specific loss functions.

Although All-In-One Face [172] provides outputs for seven different face-related tasks, we use only the facial keypoints generated by this network in our face recognition pipeline. Once we obtain the keypoints for every face in an image or a video frame, we align the faces to normalized canonical coordinates to mitigate the effects of in-plane rotation and scaling. These aligned faces are then passed to the face recognition module for subsequent processing.

To train our face representation model, we use the Universe face dataset from [20]. This is a combination of UMDFaces images [18], UMDFaces video frames [17], and curated MS-Celeb-1M [73]. The Universe dataset contains about 5.6 million images of about 58,000 identities. This includes about 3.5 million images from MS-Celeb-1M, 1.8 million video frames from UMDFaces videos, and 300,000 images from UMDFaces. This dataset has the advantage of being the union of different

datasets which makes networks trained using this dataset generalize better. Another advantage is that it contains both still images and video frames which makes the networks more robust to test datasets that contain both images and videos.

Our feature representation model is based on an Inception ResNet-v2 [198]. For pre-processing the detected faces, we crop and resize the aligned faces to each network's input dimensions. For data augmentation, we apply random horizontal flips to the input images. The Inception ResNet-v2 network is trained with the Universe dataset. This network has 244 convolution layers. We add a 512-D feature layer after these and then a final classification layer. We use crystal loss (Section 8.2) with $\alpha = 40$. The initial learning rate is set to 0.1 and is reduced by a factor of 0.2 after every $50k$ iterations. We train the network for $120k$ iterations with a batch-size of 120 on 8 NVIDIA Quadro P6000 GPUs. We resize the inputs to 299×299. We use UMDFaces [18] to train a final 128-D embedding with TPE.

For both face verification and identification, we need to compare template features. To obtain feature vectors for a template, we first average all the features for a media in the template. We further average these media-averaged features to get the final template feature.

8.2 Loss Function

The Crystal Loss [168] can be written as:

$$\begin{aligned} \text{minimize} \quad & -\frac{1}{M}\sum_{i=1}^{M}\log\frac{e^{W_{y_i}^T f(\mathbf{x}_i)+b_{y_i}}}{\sum_{j=1}^{C} e^{W_j^T f(\mathbf{x}_i)+b_j}} \\ \text{subject to} \quad & \|f(\mathbf{x}_i)\|_2 = \alpha, \quad \forall i = 1, 2, ...M, \end{aligned} \tag{8.1}$$

where $\mathbf{x}_i$ is the input image in a mini-batch of size M, y_i is the corresponding class label, $f(\mathbf{x}_i)$ is the feature descriptor obtained from the penultimate layer of DCNN, C is the number of subject classes, and W and b are the weights and bias for the last layer of the network which acts as a classifier. Equation 8.1 adds an additional L_2-constraint to the softmax loss.

The most important advantages of using Crystal Loss lie in its ability to represent each type of face with a feature of similar magnitude. This ensures that both low-quality and high-quality images are given equal weight.

8.3 Experiments

In this section, we report experimental results for end-to-end face identification and verification on four challenging evaluation datasets, *viz.*, IJB-A [107], IJB-B [216], and IJB-C [144]. We show that the proposed system achieves state-of-the-art or near results on most of the protocols.

We use ROC curves to measure the performance of face verification (1:1 matching) methods, and CMC and TPIR-FPIR curves [107] are used for evaluating face

Table 8.1: Task descriptions for IJB-A, IJB-B, and IJB-C datasets.

Task	Desciption
1:1 Verification	Verify if the given pair of templates belong to the same subject. Templates are comprised of mixed media (frames and stills).
1:N Mixed Search	Open set identification protocol using mixed media (frames and stills) as probe and two galleries G1, and G2.

identification (1:N search) in close-set and open-set settings, respectively. The IJB-A [107], IJB-B [216], and IJB-C [144] datasets contain a gallery and a probe which leads to evaluation using all positive and negative pairs. This is different from LFW [91] and YTF [217] where only a few negative pairs are used to evaluate verification performance. Another difference between LFW/YTF and the evaluation datasets here is the inclusion of templates instead of only single images. A template is a collection of images and video frames of a subject. These datasets are much more challenging than older datasets due to extreme variations in pose, illumination, and expression. Table 8.1 gives brief descriptions of the identification and verification tasks, including 1:1 verification and 1:N search.

The **IJB-B** dataset [216], which extends IJB-A, contains about $22,000$ still images and $55,000$ video frames spread over $1,845$ subjects. Evaluation is done for the same tasks as IJB-A, *viz.*, 1:1 verification, and 1:N identification. The IJB-B verification protocol consists of $8,010,270$ pairs between templates in the galleries (G1 and G2) and the probe templates. Out of these, 8 million are impostor pairs and the rest $10,270$ are genuine comparisons. Tables 8.4 and 8.5 provide the verification and identification results respectively.

The **IJB-C** evaluation dataset [144] further extends IJB-B. It contains $31,334$

Table 8.2: IJB-A Verification. The best results are in bold.

	True Accept Rate (%) @ False Accept Rate			
Method	0.0001	0.001	0.01	0.1
Casia [209]	-	51.4	73.2	89.5
Pose [6]	-	-	78.7	91.1
NAN [230]	-	88.1	94.1	97.8
3D [143]	-	72.5	88.6	-
$DCNN_{fusion}$ [35]	-	76.0	88.9	96.8
$DCNN_{tpe}$ [181]	-	81.3	90.0	96.4
$DCNN_{all}$ [172]	-	78.7	89.3	96.8
All-In-One [172]	-	82.3	92.2	97.6
TP [38]	-	-	93.9	-
$RX101_{l2+tpe}$ [169]	90.9	94.3	**97.0**	**98.4**
Ours	**91.7**	**95.3**	96.8	98.3

Table 8.3: IJB-A 1:N Mixed Search. The best results are in bold.

	TPIR (%) @ FPIR		Retrieval Rate (%)		
Method	0.01	0.1	Rank=1	Rank=5	Rank=10
Casia [209]	38.3	61.3	82.0	92.9	-
Pose [6]	52.0	75.0	84.6	92.7	94.7
BL [36]	-	-	89.5	96.3	-
NAN [230]	81.7	91.7	95.8	98	98.6
3D [143]	-	-	90.6	96.2	97.7
$DCNN_{fusion}$ [35]	65.4	83.6	94.2	98.0	98.8
$DCNN_{tpe}$ [181]	75.3	83.6	93.2	-	97.7
$DCNN_{all}$ [172]	70.4	83.6	94.1	-	98.8
All-In-One [172]	79.2	88.7	94.7	-	98.8
TP [38]	77.4	88.2	92.8	-	98.6
$RX101_{l2+tpe}$ [169]	**91.5**	95.6	97.3	-	**98.8**
Ours	91.4	**96.1**	**97.3**	**98.2**	98.5

Table 8.4: IJB-B Verification. The best results are in bold.

	True Accept Rate (%) @ False Accept Rate					
Method	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}
GOTS [216]	-	-	16.0	33.0	60.0	-
VGGFaces [157]	-	-	55.0	72.0	86.0	-
FPN [28]	-	-	83.2	91.6	96.5	-
Light CNN-29 [218]	-	-	87.7	92.0	95.3	-
VGGFace2 [26]	-	70.5	83.1	90.8	95.6	-
Center Loss [215]	31.0	63.6	80.7	90.0	95.1	98.4
MN-vc [224]	-	-	83.1	90.9	95.8	98.5
SENet50+DCN [223]	-	-	84.9	93.7	**97.5**	**99.7**
ArcFace [42]	37.5	**89.0**	**94.2**	**96.0**	97.5	98.4
Ours	27.7	61.6	89.1	94.3	97.0	98.7

Table 8.5: IJB-B 1:N Mixed Search. Note that the retrieval rates for some past methods are average over G1 and G2. The best results are in bold.

	TPIR (%) @ FPIR (For G1, G2)		Retrieval Rate (%) (For G1, G2)		
Method	0.01	0.1	Rank=1	Rank=5	Rank=10
GOTS [216]	-	-	42.0	-	62.0
VGGFace [157]	-	-	78.0	-	89.0
FPN [28]	-	-	91.1	-	96.5
Light CNN-29 [218]	-	-	91.9	94.8	-
VGGFace2 [26]	74.3	86.3	90.2	94.6	95.9
Center Loss [215]	75.5, 67.7	87.5, 82.8	92.2, 86.0	95.4, 92.5	96.2, 94.4
Ours	**83.1**, **75.5**	**93.6**, **89.3**	**95.5**, **90.8**	**97.5**, 94.2	**98.0**, **95.8**

Table 8.6: IJB-C Verification. The best results are in bold.

	True Accept Rate (%) @ False Accept Rate							
Method	10^{-8}	10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}
Center Loss [215]	36.0	37.6	66.1	78.1	85.3	91.2	95.3	98.2
MN-vc [224]	-	-	-	-	86.2	92.7	96.8	98.9
SENet50+DCN [223]	-	-	-	-	88.5	94.7	**98.3**	**99.8**
ArcFace [42]	-	-	**85.4**	**92.8**	**95.6**	**97.2**	98.0	98.8
Ours	16.5	19.5	43.6	77.6	91.9	95.6	97.8	**99.0**

Table 8.7: IJB-C 1:N Mixed Search.

	TPIR (%) @ FPIR (For G1, G2)		Retrieval Rate (%) (For G1, G2)		
Method	0.01	0.1	Rank=1	Rank=5	Rank=10
Center Loss [215]	79.1, 75.3	86.4, 84.2	91.7, 89.8	94.6, 93.6	95.6, 94.9
Ours	**87.7**, **82.4**	**93.5**, **91.0**	**95.7**, **92.8**	**97.4**, **95.4**	**97.9**, **96.4**

still images and 117,542 video frames of 3,531 subjects. There are about 20,000 genuine comparisons, and about 15.6 million impostor pairs in the verification protocol. For the 1:N mixed search protocol, there are about 20,000 probe templates. In Table 8.6 we list the results of our system for 1:1 verification. Similarly, in Table 8.7 we give results for 1:N mixed search.

8.4 Disguised Faces in the Wild

The recently announced Disguised Faces in the Wild (DFW) dataset and challenge [45,115] aims to study another covariate of the face verification pipeline - 'disguises'.

Disguises and impersonations are part of a sub-field of face recognition where the subjects are non-cooperative and are actively trying to deceive the system. A

disguise is defined as a means of altering one's appearance or concealing one's identity. This means that the subject is actively trying to adopt a new identity in order to hide his or her own. Similarly, an impersonation is the act of pretending to be another person. A subject might be trying to disguise his or her identity by adopting another identity or another person might be trying to impersonate the subject of interest.

This is a challenging face verification problem. The aim of a face verification system in such cases is to be able to identify disguises and reject impersonators. The DFW challenge [45, 115] was introduced keeping such a target in mind.

Building upon the face verification pipeline described above, we build an ensemble of two deep CNNs and achieve good preliminary results for this task. We use a large amount of data for training our models and report results on the DFW challenge test set [45, 115]. This dataset contains both UMDFaces and UMDFaces-Videos along with the MS-Celeb-1M [73] dataset. We combine a cleaned version of the MS-Celeb dataset and the UMDFaces giving us a total of 5.6 million images of about 58,000 subjects.

There are several factors to consider while designing a face verification system. One of these is the loss function used to train the deep networks. Most current methods use softmax loss for training their deep network. Softmax presents several advantages for training CNNs. It can be easily implemented using existing functions in various deep learning libraries [5, 37, 97] and does not have any restrictions on batch-size. It converges quickly. However, it is biased to the sample distribution in the training set. Unlike triplet loss, it does not specifically attend to hard samples.

Figure 8.1: Various disguises worn by Gary Oldman throughout his career as an actor. Recognizing people under disguises is clearly a challenging problem, even for humans. Designing autonomous systems for such a problem will be an important step towards complete face understanding.

It fits well on high quality data and ignores the difficult samples in the mini-batch. To overcome this limitation, Crystal Loss was introduced in [168]. It pushes samples from the same class closer and samples from different classes apart. In this work, we use the Crystal Loss for training our networks.

We build an ensemble of two deep CNNs for the task. The two networks are trained on the same dataset. We describe the architectures and training details of the two deep CNNs used in our method in Section 8.4.1 and finally, in Section 8.4.2, we report the results for the DFW challenge. An overview of a typical face verification pipeline is shown in Figure 8.2.

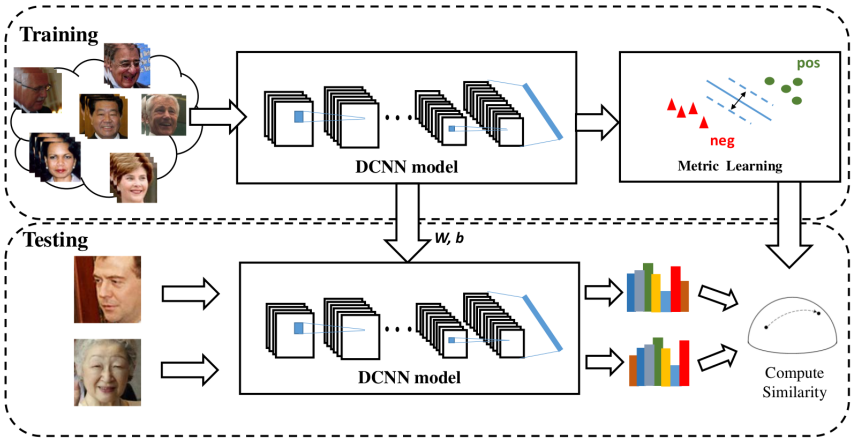


Figure 8.2: A typical face verification system pipeline. During training, a deep network is trained for classification using a large training dataset (e.g. UMDFaces [17,18], MS-Celeb-1M [73]). After training the network, a metric learning framework (e.g. triplet embedding) is used to embed the features obtained from the deep CNN into a discriminative subspace. At test time, given two faces, the features from the deep CNN are computed and embedded into the embedding subspace. Finally, a similarity score (e.g. cosine similarity) is calculated between the two embedded features.

8.4.1 Architectures

We describe the architectures of the two networks in our ensemble. We also give the training details and present the fusion algorithm for combining the outputs of the two networks.

Pre-processing: We use the All-in-one CNN [172] for face detection and alignment. We crop and resize each aligned face to each network's corresponding input size before sending them through the network. We applied a random horizontal flip as a data augmentation strategy.

ResNet-101

The first network is a ResNet-101 [84]. This network contains 101 convolutional

layers followed by a fully-connected layer of dimension 512. We use PReLU [84] activation function after every convolutional layer. In total, the training data contains $57,779$ subjects and $5,554,906$ images. The network was trained using L_2-Softmax Loss [169] with α parameter set to 50. The initial learning rate was set to 0.1, which was reduced after every 50k iterations by a factor of 0.2. The training was carried out till $250,000$ iterations with a batch size of 128. We use the Triplet Probabilistic Embedding (TPE) [181] to learn a 128-dimensional embedding using images from UMDFaces [18] dataset.

Inception ResNet-v2

We adapt the Inception-ResNet-v2 model described in Section 8.1. For final inference, we use TPE [181] to learn a 128-dimensional embedding using images from UMDFaces [18].

For fusion, we take the average of the scores obtained from the two networks as our final scores for each pair of images. More sophisticated fusion strategies will be explored in future.

8.4.2 Results

We first evaluate our approach on the relatively simple Disguised and Makeup Faces Database [214]. This dataset contains 2460 images for 410 identities. The images in this dataset are mostly celebrities with different disguises and makeups. Our method

achieves significant performance improvements over the baseline results reported in [214]. The method achieves a true accept rate (TAR) of 92% at a false accept rate (FAR) of 0.0001, and a TAR of 96.4% at FAR 0.001. This shows that our method can recognize people with make-up and disguises with high confidence.

We then evaluate our approach on the Disguised Faces in the Wild (DFW) challenge [115]. The DFW challenge provides about $7,800$ test images for about 600 identities containing both disguises and impersonations. Each identity in the test set contains a normal image, some validation images, a few images with the subject in disguise, and a few images of impersonators i.e. other people who look like the subject under consideration. The aim of this challenge is to recognize disguised faces as belonging to the subject under consideration and reject the impersonators. The challenge follows a standard face verification evaluation strategy. Each pair in the test set is assigned a similarity score by the algorithm and has an associated ground-truth label ('positive', 'negative', or 'do not care'). The evaluation criterion is a standard ROC curve which plots the True Acceptance Rate (TAR) against False Acceptance Rate (FAR).

We present results for our two networks separately and also for the combination strategy highlighted in Section 8.4.1.

Figure 8.3 shows the ROC curves for both our networks and for the final fused scores. Table 8.8 gives the TAR values at $FAR = 0.01$ and $FAR = 0.001$ for our fused model and compares it with recent works [116].

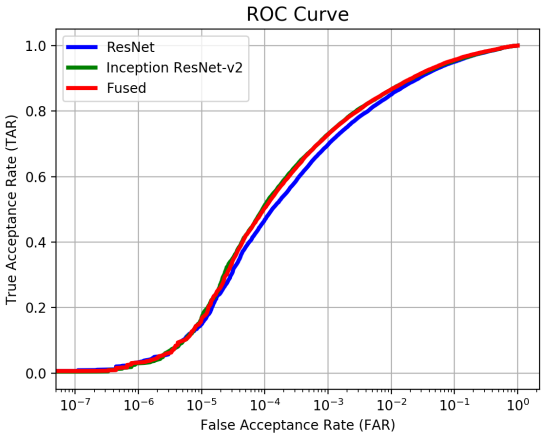


Figure 8.3: Results ROC curve. The plot shows the performance (TAR) of our two networks and the fused features at different False Acceptance Rates (FAR) for the DFW dataset.

8.5 Discussion and Conclusion

In this chapter, we presented a fast and high-performance system for face recognition. We used the Inception ResNet-v2 network as the feature backbone for our model. We use a large dataset comprising of our UMDFaces datasets and the MSCeleb dataset to train the network with Crystal loss. We showed that such a model can achieve near state-of-the-art results for several challenging face recognition benchmarks like IJB-A, IJB-B, and IJB-C. We further showed that an ensemble of our model and a ResNet trained in a similar fashion can achieve good performance on the recent Disguised Faces in the Wild challenge. This demonstrates the importance of training deep networks using our datasets and Crystal loss. The feature obtained from these networks are effective for face verification and identification.

Table 8.8: TAR (%) at different FAR values for the three different features used in this work for the DFW dataset. The performance in bold is the best in each column and the underlined numbers are the second best in a column.

Feature	$FAR = 0.001$	$FAR = 0.01$
AEFRL	63.52	80.52
VGG-Face	17.73	33.76
ByteFace	54.16	75.53
DDRNET	49.08	71.43
DisguiseNet	23.25	60.89
LearnedSiamese	18.79	39.73
MEDC	63.22	81.31
MiRA-Face	**75.08**	**89.04**
Ours	<u>68.52</u>	<u>83.49</u>

Chapter 9: Summary and Suggestions for Future Work

In this dissertation, we focused on improving the recognition of humans, objects, and their interactions. These are constituent parts of the general problem of scene understanding. A better understanding of a scene can be gained by asking and answering questions. Therefore, in this work, we also address the problem of visual question answering. We summarize some of our most important contributions and observations here.

We started by introducing ZSD as a novel problem in computer vision (Chapter 3). We discussed the challenges associated with ZSD and proposed several approaches for dealing with these challenges. In particular, we proposed two background-aware approaches: 1.) statistically assigned background method; and 2.) latent assignment based method. We also proposed the densely sampled embedding space method to deal with the issue of sparse distribution of classes in the semantic space. We showed that ZSD is a challenging problem which provides several avenues for research. We also demonstrated that our proposed approaches can give good performance improvements over baselines.

In the next two chapters (Chapters 4 and 5), we proposed two approaches for HOI detection. The first of these, called Functional Generalization, was based on

the idea that human-object interactions look similar for functionally similar objects. We proposed a data augmentation strategy based on this insight. We showed that a large amount of additional data can be generated using the existing labeled data and exploiting functional similarities between objects. We proposed a simple model which could utilize the additional data and obtain state-of-the-art performance for HOI detection.

Our second HOI detection model was explicitly designed to exploit the relative locations and orientations of the human and the object in the scene. The proposed model comprised of a layout module and a visual module. We also introduced the idea of spatial priming, where the prediction from the layout module is used to prime the visual module. We showed that this model is able to achieve an even higher performance than the Functional Generalization model demonstrating the effectiveness of utilizing the spatial information.

In Chapter 6, we introduced the generalized VQA problem of ISVQA which is a challenging setting requiring reasoning over several images from a scene together. We collected an annotated dataset for ISVQA. We provided an exhaustive analysis of the dataset and proposed several VQA-based baseline methods. We observed that even these strong baseline methods are not sufficient for ISVQA. This revealed the unique challenges associated with ISVQA and showed that this problem cannot be solved trivially.

Finally, in Chapters 7 and 8, we described two large-scale datasets and a fast and accurate pipeline for face identification and verification. We explored several important questions that should be considered before designing any face recognition

system and showed that carefully answering those questions is essential for designing high-performance face recognition systems. We showed that using the UMDFaces datasets and Crystal loss, we can design face recognition pipelines that can beat state-of-the-art methods on the IJB-A, IJB-B, IJB-C, and DFW benchmarks.

9.1 Future Work

Most of the accomplished work presented in this dissertation has focused on scene understanding from still images. We have not paid much attention to video understanding. Future work should focus on using additional semantic information for video understanding. Future work can also try to improve human-object interaction detection by using additional semantic knowledge from artwork datasets like the Flintstones dataset [77].

9.1.1 Frame Semantics for Video Understanding

Most current video and action recognition methods [27,51] rely completely on pixel-level visual data for making an inference. However, reasoning on the objects present in a scene can provide important cues about the setting or action in the video. An event or an activity can be understood on the basis of the description and interactions of the participants in it. For example, the action of `cooking` involves a cook (person), the food, a container, and a heating instrument. If we detect a person, a stove, a pan, and some vegetables, and we know the typical purpose of each object, we can infer that the scene probably involves `cooking`.

An activity can be understood as a *frame* [2, 16, 145] and the participant as *frame elements*. The frame problem is a classical problem in AI which deals with the ways of writing the effects of an action in logic formulae [3]. Certain words can *evoke* a frame. These words can be considered as specific instances of a general activity. This structure can be used for video recognition. Frame-level annotations from the FrameNet [2] lexical database can provide important cues.

9.1.2 Annotated Artwork for Human-Object Interaction Detection

Labeled training data is still a major issue in human-object interaction detection. We believe that more human-object layout data can greatly benefit our Spatial Priming model. Recently, Gupta *et al.* [77] released the Flintstones dataset which contains annotations for several human-human and human-object interactions. The authors utilized scripts and character locations to generate this data. We propose to use the layout information from this dataset to further improve our models.

In addition, we also observed that the HICO-Det dataset [30] contains inexhaustive labels. Therefore, some correct HOI detections might be marked as incorrect. Since HOI detection methods are evaluated in mean average precision (mAP), missing labels have a negative impact on the performance numbers. Future work should quantify this effect by getting a part of the HICO-Det dataset exhaustively annotated. This will give a better understanding of the performance of various models and the avenues for improvement.

Further, future research should focus on improving performance for rare classes.

Clever class-weighing strategies and using more semantic knowledge as in [160] could be some ways of going forward. Another limitation of our method is the dependence on a pre-trained object detector. Future work should also focus on jointly training the HOI prediction model and the object detector. Since HOI detection and object detection have complementary objectives (a better object detector leads to better HOI detection), this line of approach could significantly improve performance for both HOI detection and object detection.

9.1.3 Lexical Ontology and Hierarchical Prediction for ZSD

For ZSD, it is important to incorporate some lexical ontology information ("is a" and "is part of" relationships) during training and testing for learning models on large vocabularies. Most existing object detection frameworks ignore the hierarchical nature of object classes. For example, a "cat" should incur a lower loss when predicted as "animal" vs. when predicted as "vehicle". Although few works have tried to address this issue [79, 173], we believe further work in this direction would be beneficial for zero-shot detection. We also feel that additional work is needed to generalize bounding-box regression and hard-negative mining for new objects.

Bibliography

[1] Available at `http://places2.csail.mit.edu`.

[2] Available at `https://framenet.icsi.berkeley.edu/fndrupal/`.

[3] Available at `https://plato.stanford.edu/entries/frame-problem/`.

[4] youtube-dl `https://github.com/rg3/youtube-dl`. online; accessed 06-march-2017.

[5] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[6] W. AbdAlmageed, Y. Wu, S. Rawls, S. Harel, T. Hassne, I. Masi, J. Choi, J. Lekust, J. Kim, P. Natarajana, R. Nevatia, and G. Medioni. Face recognition using deep multi-pose representations. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016.

[7] M. Acharya, K. Kafle, and C. Kanan. TallyQA: Answering Complex Counting Questions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:8076–8084, 2019.

[8] A. Agrawal, D. Batra, and D. Parikh. Analyzing the Behavior of Visual Question Answering Models. *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1955–1960, 2016.

[9] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for attribute-based classification. In *CVPR*, pages 819–826. IEEE, 2013.

[10] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele. Evaluation of output embeddings for fine-grained image classification. In *CVPR*, pages 2927–2936. IEEE, 2015.

[11] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 6077–6086, 2018.

[12] L. Anne Hendricks, K. Burns, K. Saenko, T. Darrell, and A. Rohrbach. Women also snowboard: Overcoming bias in captioning models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 771–787, 2018.

[13] L. Anne Hendricks, S. Venugopalan, M. Rohrbach, R. Mooney, K. Saenko, T. Darrell, J. Mao, J. Huang, A. Toshev, O. Camburu, et al. Deep compositional captioning: Describing novel object categories without paired training data. In *CVPR*, pages 1–10. IEEE, 2016.

[14] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: Visual Question Answering. *Proceedings - IEEE International Conference on Computer Vision, ICCV*, 2015.

[15] Y. Aytar, C. Vondrick, and A. Torralba. See, hear, and read: Deep aligned representations. *arXiv preprint arXiv:1706.00932*, 2017.

[16] C. F. Baker, C. J. Fillmore, and J. B. Lowe. The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics, 1998.

[17] A. Bansal, C. Castillo, R. Ranjan, and R. Chellappa. The do's and don'ts for cnn-based face verification. *arXiv preprint arXiv:1705.07426*, 5, 2017.

[18] A. Bansal, A. Nanduri, C. Castillo, R. Ranjan, and R. Chellappa. Umdfaces: An annotated face dataset for training deep networks. *arXiv preprint arXiv:1611.01484*, 2016.

[19] A. Bansal, S. S. Rambhatla, A. Shrivastava, and R. Chellappa. Detecting human-object interactions via functional generalization. *arXiv preprint arXiv:1904.03181*, 2019.

[20] A. Bansal, R. Ranjan, C. D. Castillo, and R. Chellappa. Deep features for recognizing disguised faces in the wild. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 10–106. IEEE, 2018.

[21] A. Bansal, K. Sikka, G. Sharma, R. Chellappa, and A. Divakaran. Zero-shot object detection. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[22] A. Bansal and K. Venkatesh. People counting in high density crowds from still images. *arXiv preprint arXiv:1507.08445*, 2015.

[23] P. N. Belhumeur, D. W. Jacobs, D. J. Kriegman, and N. Kumar. Localizing parts of faces using a consensus of exemplars. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 545–552, 2011.

[24] A. Bendale and T. E. Boult. Towards open set deep networks. In *CVPR*, pages 1563–1572. IEEE, 2016.

[25] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuScenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019.

[26] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *Automatic Face & Gesture Recognition (FG 2018), 2018 13th IEEE International Conference on*, pages 67–74. IEEE, 2018.

[27] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[28] F.-J. Chang, A. T. Tran, T. Hassner, I. Masi, R. Nevatia, and G. Medioni. Faceposenet: Making a case for landmark-free face alignment. In *Computer Vision Workshop (ICCVW), 2017 IEEE International Conference on*, pages 1599–1608. IEEE, 2017.

[29] S. Changpinyo, W.-L. Chao, B. Gong, and F. Sha. Synthesized classifiers for zero-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5327–5336, 2016.

[30] Y.-W. Chao, Y. Liu, X. Liu, H. Zeng, and J. Deng. Learning to detect human-object interactions. *arXiv preprint arXiv:1702.05448*, 2017.

[31] Y.-W. Chao, Z. Wang, Y. He, J. Wang, and J. Deng. Hico: A benchmark for recognizing human-object interactions in images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1017–1025, 2015.

[32] B. Chen, W. Deng, and J. Du. Noisy softmax: Improving the generalization ability of dcnn via postponing the early softmax saturation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[33] D. Chen, G. Hua, F. Wen, and J. Sun. Supervised transformer network for efficient face detection. In *European Conference on Computer Vision*, pages 122–138. Springer, 2016.

[34] J.-C. Chen, V. M. Patel, and R. Chellappa. Unconstrained face verification using deep cnn features. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–9. IEEE, 2016.

[35] J.-C. Chen, V. M. Patel, and R. Chellappa. Unconstrained face verification using deep cnn features. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016.

[36] A. R. Chowdhury, T.-Y. Lin, S. Maji, and E. Learned-Miller. One-to-many face recognition with bilinear CNNs. In *IEEE Winter Conference on Applications of Computer Vision (WACV).*, pages 1–9. IEEE, 2016.

[37] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS workshop*, number EPFL-CONF-192376, 2011.

[38] N. Crosswhite, J. Byrne, O. M. Parkhi, C. Stauffer, Q. Cao, and A. Zisserman. Template adaptation for face verification and identification. *arXiv preprint arXiv:1603.03958*, 2016.

[39] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. 1:886–893 vol. 1, June 2005.

[40] V. Delaitre, J. Sivic, and I. Laptev. Learning person-object interactions for action recognition in still images. In *Advances in neural information processing systems*, pages 1503–1511, 2011.

[41] B. Demirel, R. G. Cinbis, and N. Ikizler-Cinbis. Zero-shot object detection by hybrid region embedding. *arXiv preprint arXiv:1805.06157*, 2018.

[42] J. Deng, J. Guo, and S. Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *arXiv preprint arXiv:1801.07698*, 2018.

[43] C. Desai and D. Ramanan. Detecting actions, poses, and objects with relational phraselets. In *European Conference on Computer Vision*, pages 158–172. Springer, 2012.

[44] M. T. Desta, L. Chen, and T. Kornuta. Object-based reasoning in VQA. *Proceedings - 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018*, 2018-January:1814–1823, 2018.

[45] T. I. Dhamecha, R. Singh, M. Vatsa, and A. Kumar. Recognizing disguised faces: Human and machine evaluation. *PloS one*, 9(7):e99212, 2014.

[46] M. Elhoseiny, B. Saleh, and A. Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *CVPR*, pages 2584–2591. IEEE, 2013.

[47] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

[48] F. Faghri, D. J. Fleet, J. R. Kiros, and S. Fidler. Vse++: Improved visual-semantic embeddings. *arXiv preprint arXiv:1707.05612*, 2017.

[49] T. C. Faltemier, K. W. Bowyer, and P. J. Flynn. Using a multi-instance enrollment representation to improve 3d face recognition. In *Biometrics: Theory, Applications, and Systems, 2007. BTAS 2007. First IEEE International Conference on*, pages 1–6. IEEE, 2007.

[50] H.-S. Fang, J. Cao, Y.-W. Tai, and C. Lu. Pairwise body-part attention for recognizing human-object interactions. *arXiv preprint arXiv:1807.10889*, 2018.

[51] C. Feichtenhofer, H. Fan, J. Malik, and K. He. Slowfast networks for video recognition. *arXiv preprint arXiv:1812.03982*, 2018.

[52] C. Feichtenhofer, A. Pinz, and R. Wildes. Spatiotemporal residual networks for video action recognition. In *Advances in neural information processing systems*, pages 3468–3476, 2016.

[53] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, Sept 2010.

[54] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Pose search: retrieving people using their pose. In *CVPR*, pages 1–8. IEEE, 2009.

[55] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, and T. Mikolov. Devise: A deep visual-semantic embedding model. In *NIPS*, pages 2121–2129, 2013.

[56] Y. Fu, T. M. Hospedales, T. Xiang, and S. Gong. Attribute learning for understanding unstructured social activity. In *ECCV*, pages 530–543. Springer, 2012.

[57] Y. Fu, T. Xiang, Y.-G. Jiang, X. Xue, L. Sigal, and S. Gong. Recent advances in zero-shot recognition. *arXiv preprint arXiv:1710.04837*, 2017.

[58] Y. Fu, Y. Yang, T. Hospedales, T. Xiang, and S. Gong. Transductive multi-label zero-shot learning. *arXiv preprint arXiv:1503.07790*, 2015.

[59] C. Gao, Y. Zou, and J.-B. Huang. ican: Instance-centric attention network for human-object interaction detection. *arXiv preprint arXiv:1808.10437*, 2018.

[60] P. Gao, Z. Jiang, H. You, P. Lu, S. Hoi, X. Wang, and H. Li. Dynamic Fusion with Intra- and Inter-modality Attention Flow for Visual Question Answering. *Proceedings - IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2019.

[61] P. Gao, H. You, Z. Zhang, X. Wang, and H. Li. Multi-modality Latent Interaction Network for Visual Question Answering. *International Conference on Computer Vision, ICCV*, 2019.

[62] S. Gavves, T. Mensink, T. Tommasi, C. Snoek, and T. Tuytelaars. Active transfer learning with zero-shot priors: Reusing past datasets for future tasks. In *ICCV*, pages 2731–2739. IEEE, 2015.

[63] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[64] J. Gibson. The theory of affordances the ecological approach to visual perception (pp. 127-143), 1979.

[65] R. Girdhar and D. Ramanan. Attentional pooling for action recognition. In *Advances in Neural Information Processing Systems*, pages 34–45, 2017.

[66] R. Girshick. Fast r-cnn. In *ICCV*, pages 1440–1448. IEEE, 2015.

[67] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Region-based convolutional networks for accurate object detection and segmentation. *TPAMI*, 38(1):142–158, 2016.

[68] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron. `https://github.com/facebookresearch/detectron`, 2018.

[69] G. Gkioxari, R. Girshick, P. Dollár, and K. He. Detecting and recognizing human-object interactions. *arXiv preprint arXiv:1704.07333*, 2017.

[70] D. Gordon, A. Kembhavi, M. Rastegari, J. Redmon, D. Fox, and A. Farhadi. IQA: Visual Question Answering in Interactive Environments. *Proceedings - IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4089–4098, 2018.

[71] Y. Goyal, T. Khot, A. Agrawal, D. Summers-Stay, D. Batra, and D. Parikh. Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering. *International Journal of Computer Vision*, 127(4):398–414, apr 2019.

[72] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. Ms-celeb-1m: Challenge of recognizing one million celebri-ties in the real world.

[73] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *European Conference on Computer Vision*, pages 87–102. Springer, 2016.

[74] A. Gupta and L. S. Davis. Objects in action: An approach for combining action understanding and object perception. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[75] A. Gupta, A. Kembhavi, and L. S. Davis. Observing human-object interactions: Using spatial and functional compatibility for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1775–1789, 2009.

[76] S. Gupta and J. Malik. Visual semantic role labeling. *arXiv preprint arXiv:1505.04474*, 2015.

[77] T. Gupta, D. Schwenk, A. Farhadi, D. Hoiem, and A. Kembhavi. Imagine this! scripts to compositions to videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 598–613, 2018.

[78] T. Gupta, A. Schwing, and D. Hoiem. No-frills human-object interaction detection: Factorization, appearance and layout encodings, and training techniques. *arXiv preprint arXiv:1811.05967*, 2018.

[79] T. Gupta, K. Shih, S. Singh, and D. Hoiem. Aligned image-word representations improve inductive transfer across vision-language tasks. *arXiv preprint arXiv:1704.00260*, 2017.

[80] D. Gurari, Q. Li, A. J. Stangl, A. Guo, C. Lin, K. Grauman, J. Luo, and J. P. Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3608–3617, 2018.

[81] E. M. Hand, C. Castillo, and R. Chellappa. Doing the best we can with what we have: Multi-label balancing with selective learning for attribute prediction. In *AAAI Conference on Artificial Intelligence*. AAAI, 2018.

[82] T. Hassner, S. Harel, E. Paz, and R. Enbar. Effective face frontalization in unconstrained images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4295–4304, 2015.

[83] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[84] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[85] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[86] D. Hoiem, Y. Chodpathumwan, and Q. Dai. Diagnosing error in object detectors. In *ECCV*, pages 340–353. Springer, 2012.

[87] G. Hu, X. Peng, Y. Yang, T. Hospedales, and J. Verbeek. Frankenstein: Learning deep face representations using small data. *arXiv preprint arXiv:1603.06470*, 2016.

[88] J.-F. Hu, W.-S. Zheng, J. Lai, S. Gong, and T. Xiang. Recognising human-object interaction via exemplar based modelling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3144–3151, 2013.

[89] P. Hu and D. Ramanan. Finding tiny faces. *arXiv preprint arXiv:1612.04402*, 2016.

[90] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Weinberger. Deep networks with stochastic depth. *arXiv preprint arXiv:1603.09382*, 2016.

[91] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[92] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *IEEE CVPR*, volume 4, 2017.

[93] D. A. Hudson and C. D. Manning. GQA: A New Dataset for Real-World Visual Reasoning and Compositional Question Answering. *Proceedings - IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2019.

[94] L. P. Jain, W. J. Scheirer, and T. E. Boult. Multi-class open set recognition using probability of inclusion. In *ECCV*, pages 393–409. Springer, 2014.

[95] V. Jain and E. G. Learned-Miller. Fddb: A benchmark for face detection in unconstrained settings. *UMass Amherst Technical Report*, 2010.

[96] Y. Jang, Y. Song, Y. Yu, Y. Kim, and G. Kim. TGIF-QA: Toward spatio-temporal reasoning in visual question answering. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, pages 1359–1367. Institute of Electrical and Electronics Engineers Inc., nov 2017.

[97] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

[98] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2901–2910, 2017.

[99] J. Johnson, R. Krishna, M. Stark, L.-J. Li, D. Shamma, M. Bernstein, and L. Fei-Fei. Image retrieval using scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3668–3678, 2015.

[100] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[101] D. Kahneman and P. Egan. *Thinking, fast and slow*, volume 1. Farrar, Straus and Giroux New York, 2011.

[102] N. D. Kalka, B. Maze, J. A. Duncan, K. O'Connor, S. Elliott, K. Hebert, J. Bryan, and A. K. Jain. Ijb–s: Iarpa janus surveillance video benchmark. In *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–9. IEEE, 2018.

[103] K. Kato, Y. Li, and A. Gupta. Compositional learning for human object interaction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 234–251, 2018.

[104] V. Kazemi and J. Sullivan. One millisecond face alignment with an ensemble of regression trees. In *CVPR*, 2014.

[105] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[106] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[107] B. F. Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, M. Burge, and A. K. Jain. Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1931–1939. IEEE, 2015.

[108] E. Kodirov, T. Xiang, and S. Gong. Semantic autoencoder for zero-shot learning. *arXiv preprint arXiv:1704.08345*, 2017.

[109] A. Kolesnikov, C. H. Lampert, and V. Ferrari. Detecting visual relationships using box attention. *arXiv preprint arXiv:1807.02136*, 2018.

[110] M. Köstinger, P. Wohlhart, P. M. Roth, and H. Bischof. Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 2144–2151. IEEE, 2011.

[111] I. Krasin, T. Duerig, N. Alldrin, V. Ferrari, S. Abu-El-Haija, A. Kuznetsova, H. Rom, J. Uijlings, S. Popov, A. Veit, S. Belongie, V. Gomes, A. Gupta, C. Sun, G. Chechik, D. Cai, Z. Feng, D. Narayanan, and K. Murphy. Open-images: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*, 2017.

[112] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017.

[113] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[114] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 365–372. IEEE, 2009.

[115] V. Kushwaha, M. Singh, R. Singh, M. Vatsa, N. Ratha, and R. Chellappa. Disguised faces in the wild. Technical report, 2018.

[116] V. Kushwaha, M. Singh, R. Singh, M. Vatsa, N. Ratha, and R. Chellappa. Disguised faces in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, volume 8, 2018.

[117] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, pages 951–958. IEEE, 2009.

[118] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465, 2014.

[119] E. Learned-Miller, G. B. Huang, A. RoyChowdhury, H. Li, and G. Hua. Labeled faces in the wild: A survey. In *Advances in Face Detection and Facial Image Analysis*, pages 189–248. Springer, 2016.

[120] J. Lei, L. Yu, M. Bansal, and T. L. Berg. TVQA: Localized, Compositional Video Question Answering. *Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[121] J. Lei, L. Yu, T. L. Berg, and M. Bansal. TVQA+: Spatio-Temporal Grounding for Video Question Answering. *arXiv preprint arXiv:1904.11574*, 2019.

[122] G. Levi and T. Hassner. Age and gender classification using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 34–42, 2015.

[123] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua. A convolutional neural network cascade for face detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5325–5334, 2015.

[124] Y.-L. Li, S. Zhou, X. Huang, L. Xu, Z. Ma, H.-S. Fang, Y.-F. Wang, and C. Lu. Transferable interactiveness prior for human-object interaction detection. *arXiv preprint arXiv:1811.08264*, 2018.

[125] Z. Li, E. Gavves, T. Mensink, and C. G. Snoek. Attributes make sense on segmented objects. In *ECCV*, pages 350–365. Springer, 2014.

[126] J. Liang, L. Jiang, L. Cao, L. J. Li, and A. Hauptmann. Focal Visual-Text Attention for Visual Question Answering. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 6135–6143, 2018.

[127] J. J. Lim, R. R. Salakhutdinov, and A. Torralba. Transfer learning by borrowing examples for multiclass object detection. In *NIPS*, pages 118–126, 2011.

[128] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[129] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[130] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.

[131] X. Lin and D. Parikh. Don't Just Listen, Use Your Imagination: Leveraging Visual Common Sense for Non-Visual Tasks. *Proceedings - IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2015.

[132] J. Liu, Y. Deng, T. Bai, Z. Wei, and C. Huang. Targeting ultimate accuracy: Face recognition via deep embedding. *arXiv preprint arXiv:1506.07310*, 2015.

[133] J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attributes. In *CVPR*, pages 3337–3344. IEEE, 2011.

[134] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*, pages 21–37. Springer, 2016.

[135] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition. *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[136] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[137] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei. Visual relationship detection with language priors. In *ECCV*, pages 852–869. Springer, 2016.

[138] J. Lu, J. Yang, D. Batra, and D. Parikh. Hierarchical question-image co-attention for visual question answering. In *Advances In Neural Information Processing Systems*, pages 289–297, 2016.

[139] A. Mallya and S. Lazebnik. Learning models for actions and person-object interactions with transfer to question answering. In *European Conference on Computer Vision*, pages 414–428. Springer, 2016.

[140] K. Marino, M. Rastegari, A. Farhadi, and R. Mottaghi. OK-VQA : A Visual Question Answering Benchmark Requiring External Knowledge. *Proceedings - IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 3195–3204, 2019.

[141] I. Masi, T. Hassner, A. T. Tran, and G. Medioni. Rapid synthesis of massive face sets for improved face recognition. In *Automatic Face & Gesture Recognition (FG 2017), 2017 12th IEEE International Conference on*, pages 604–611. IEEE, 2017.

[142] I. Masi, S. Rawls, G. Medioni, and P. Natarajan. Pose-aware face recognition in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4838–4846, 2016.

[143] I. Masi, A. T. Tran, T. Hassner, J. T. Leksut, and G. Medioni. Do we really need to collect millions of faces for effective face recognition? In *European Conference on Computer Vision*, pages 579–596. Springer, 2016.

[144] B. Maze, J. Adams, J. A. Duncan, N. Kalka, T. Miller, C. Otto, A. K. Jain, W. T. Niggel, J. Anderson, J. Cheney, et al. Iarpa janus benchmark–c: Face dataset and protocol. In *11th IAPR International Conference on Biometrics*, 2018.

[145] J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In *Readings in artificial intelligence*, pages 431–450. Elsevier, 1981.

[146] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[147] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[148] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[149] M. Najibi, P. Samangouei, R. Chellappa, and L. Davis. Ssh: Single stage headless face detector. *arXiv preprint arXiv:1708.03979*, 2017.

[150] H. Nam, J.-W. Ha, and J. Kim. Dual attention networks for multimodal reasoning and matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 299–307, 2017.

[151] A. Nech and I. Kemelmacher-Shlizerman. Megaface 2: 672,057 identities for face recognition. 2016.

[152] H.-W. Ng and S. Winkler. A data-driven approach to cleaning large face datasets. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 343–347. IEEE, 2014.

[153] H. Noh, T. Kim, J. Mun, and B. Han. Transfer Learning via Unsupervised Task Discovery for Visual Question Answering. *Proceedings - IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2019.

[154] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean. Zero-shot learning by convex combination of semantic embeddings. *arXiv preprint arXiv:1312.5650*, 2013.

[155] D. Parikh, A. Kovashka, A. Parkash, and K. Grauman. Relative attributes for enhanced human-machine communication. In *AAAI*, 2012.

[156] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, volume 1, page 6, 2015.

[157] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, volume 1, page 6, 2015.

[158] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[159] J. Peyre, I. Laptev, C. Schmid, and J. Sivic. Weakly-supervised learning of visual relations. *arXiv preprint arXiv:1707.09472*, 2017.

[160] J. Peyre, I. Laptev, C. Schmid, and J. Sivic. Detecting rare visual relations using analogies. *arXiv preprint arXiv:1812.05736*, 2018.

[161] G.-J. Qi, C. Aggarwal, Y. Rui, Q. Tian, S. Chang, and T. Huang. Towards cross-category knowledge propagation for learning visual concepts. In *CVPR*, pages 897–904. IEEE, 2011.

[162] S. Qi, W. Wang, B. Jia, J. Shen, and S.-C. Zhu. Learning human-object interactions by graph parsing neural networks. *arXiv preprint arXiv:1808.07962*, 2018.

[163] X. Qi and L. Zhang. Face recognition via centralized coordinate learning. *arXiv preprint arXiv:1801.05678*, 2018.

[164] R. Qiao, L. Liu, C. Shen, and A. v. d. Hengel. Visually aligned word embeddings for improving zero-shot learning. *arXiv preprint arXiv:1707.05427*, 2017.

[165] S. Rahman, S. Khan, and F. Porikli. Zero-shot object detection: Learning to simultaneously recognize and localize novel concepts. *arXiv preprint arXiv:1803.06049*, 2018.

[166] S. Rahman, S. H. Khan, and F. Porikli. A unified approach for conventional zero-shot, generalized zero-shot and few-shot learning. *arXiv preprint arXiv:1706.08653*, 2017.

[167] R. Ranjan, A. Bansal, H. Xu, S. Sankaranarayanan, J.-C. Chen, C. D. Castillo, and R. Chellappa. Crystal loss and quality pooling for unconstrained face verification and recognition. *arXiv preprint arXiv:1804.01159*, 2018.

[168] R. Ranjan, A. Bansal, J. Zheng, H. Xu, J. Gleason, B. Lu, A. Nanduri, J.-C. Chen, C. D. Castillo, and R. Chellappa. A fast and accurate system for face detection, identification, and verification. *arXiv preprint arXiv:1809.07586*, 2018.

[169] R. Ranjan, C. D. Castillo, and R. Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017.

[170] R. Ranjan, V. M. Patel, and R. Chellappa. A deep pyramid deformable part model for face detection. In *Biometrics Theory, Applications and Systems (BTAS), 2015 IEEE 7th International Conference on*, pages 1–8. IEEE, 2015.

[171] R. Ranjan, V. M. Patel, and R. Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *arXiv preprint arXiv:1603.01249*, 2016.

[172] R. Ranjan, S. Sankaranarayanan, C. Castillo, and R. Chellappa. An all-in-one convolutional neural network for face analysis. *arXiv preprint arXiv:1611.00851*, 2016.

[173] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788. IEEE, 2016.

[174] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016.

[175] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015.

[176] R. Rothe, R. Timofte, and L. V. Gool. Dex: Deep expectation of apparent age from a single image. In *IEEE International Conference on Computer Vision Workshop on ChaLearn Looking at People*, pages 10–15, 2015.

[177] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[178] M. A. Sadeghi and A. Farhadi. Recognition using visual phrases. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1745–1752. IEEE, 2011.

[179] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 397–403, 2013.

[180] S. Saito, T. Li, and H. Li. Real-time facial segmentation and performance capture from rgb input. In *European Conference on Computer Vision*, pages 244–261. Springer, 2016.

[181] S. Sankaranarayanan, A. Alavi, and R. Chellappa. Triplet similarity embedding for face verification. *arXiv preprint arXiv:1602.03418*, 2016.

[182] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap. A simple neural network module for relational reasoning. *Advances in Neural Information Processing Systems*, 2017-December(Nips):4968–4977, 2017.

[183] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra. Habitat: A Platform for Embodied AI Research. *arXiv preprint arXiv:1904.01201*, 2019.

[184] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[185] M. Seeger. Learning with labeled and unlabeled data. Technical report, 2000.

[186] G. Sharma, F. Jurie, and C. Schmid. Expanded parts model for semantic description of humans in still images. *TPAMI*, 39(1):87–101, 2017.

[187] L. Shen, S. Yeung, J. Hoffman, G. Mori, and L. Fei-Fei. Scaling human-object interaction recognition through zero-shot learning. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1568–1576. IEEE, 2018.

[188] A. Shrivastava and A. Gupta. Contextual priming and feedback for faster r-cnn. In *European Conference on Computer Vision*, pages 330–348. Springer, 2016.

[189] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta. Beyond skip connections: Top-down modulation for object detection. *arXiv preprint arXiv:1612.06851*, 2016.

[190] V. A. Sindagi and V. M. Patel. Cnn-based cascaded multi-task learning of high-level prior and density estimation for crowd counting. In *Advanced Video and Signal Based Surveillance (AVSS), 2017 14th IEEE International Conference on*, pages 1–6. IEEE, 2017.

[191] A. Singh, V. Natarajan, Y. Jiang, X. Chen, M. Shah, M. Rohrbach, D. Batra, and D. Parikh. Pythia-a platform for vision & language research. In *SysML Workshop, NeurIPS*, volume 2018, 2018.

[192] A. Singh, V. Natarajan, M. Shah, Y. Jiang, X. Chen, D. Batra, D. Parikh, and M. Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[193] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng. Zero-shot learning through cross-modal transfer. In *NIPS*, pages 935–943, 2013.

[194] Y. Sun, Y. Chen, X. Wang, and X. Tang. Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems*, pages 1988–1996, 2014.

[195] Y. Sun, D. Liang, X. Wang, and X. Tang. Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, 2015.

[196] Y. Sun, X. Wang, and X. Tang. Hybrid deep learning for face verification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1489–1496, 2013.

[197] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1891–1898, 2014.

[198] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, pages 4278–4284, 2017.

[199] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[200] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Web-scale training for face identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2746–2754, 2015.

[201] M. Tapaswi, Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun, and S. Fidler. MovieQA: Understanding Stories in Movies through Question-Answering. *Proceedings - IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, dec 2016.

[202] A. Torralba and A. A. Efros. Unbiased look at dataset bias. 2011.

[203] A. Trott, C. Xiong, and R. Socher. Interpretable counting for visual question answering. In *International Conference on Learning Representations*, 2018.

[204] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[205] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001.

[206] P. Viola and M. J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[207] P. Viola and M. J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[208] B. Wan, D. Zhou, Y. Liu, R. Li, and X. He. Pose-aware multi-level feature network for human object interaction detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[209] D. Wang, C. Otto, and A. K. Jain. Face search at scale: 80 million gallery. *arXiv preprint arXiv:1507.07242*, 2015.

[210] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5265–5274, 2018.

[211] N. Wang, X. Gao, D. Tao, H. Yang, and X. Li. Facial feature point detection: A comprehensive survey. *Neurocomputing*, 2017.

[212] P. Wang, Q. Wu, C. Shen, and A. V. D. Hengel. The VQA-Machine : Learning How to Use Existing Vision Algorithms. *Proceedings - IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1173–1182, 2017.

[213] T. Wang, R. M. Anwer, M. H. Khan, F. S. Khan, Y. Pang, L. Shao, and J. Laaksonen. Deep contextual attention for human-object interaction detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[214] T. Y. Wang and A. Kumar. Recognizing human faces under disguise and makeup. In *Identity, Security and Behavior Analysis (ISBA), 2016 IEEE International Conference on*, pages 1–7. IEEE, 2016.

[215] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pages 499–515. Springer, 2016.

[216] C. Whitelam, E. Taborsky, A. Blanton, B. Maze, J. C. Adams, T. Miller, N. D. Kalka, A. K. Jain, J. A. Duncan, K. Allen, et al. Iarpa janus benchmark-b face dataset. In *CVPR Workshops*, volume 2, page 6, 2017.

[217] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 529–534. IEEE, 2011.

[218] X. Wu, R. He, Z. Sun, and T. Tan. A light cnn for deep face representation with noisy labels. *IEEE Transactions on Information Forensics and Security*, 13(11):2884–2896, 2018.

[219] Y. Wu, H. Liu, J. Li, and Y. Fu. Deep face recognition with center invariant loss. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pages 408–414. ACM, 2017.

[220] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9068–9079, 2018.

[221] Y. Xian, Z. Akata, G. Sharma, Q. Nguyen, M. Hein, and B. Schiele. Latent embeddings for zero-shot classification. In *CVPR*, pages 69–77. IEEE, 2016.

[222] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata. Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. *arXiv preprint arXiv:1707.00600*, 2017.

[223] W. Xie, L. Shen, and A. Zisserman. Comparator networks. In *European Conference on Computer Vision*, pages 811–826. Springer, 2018.

[224] W. Xie and A. Zisserman. Multicolumn networks for face recognition. *arXiv preprint arXiv:1807.09192*, 2018.

[225] L. Xiong, K. Jayashree, J. Zhao, J. Feng, S. Pranata, and S. Shen. A good practice towards top performance of face recognition: Transferred deep feature fusion. *arXiv preprint arXiv:1704.00438*, 2017.

[226] B. Xu, Y. Fu, Y.-G. Jiang, B. Li, and L. Sigal. Heterogeneous knowledge transfer in video emotion recognition, attribution and summarization. *TPAMI*, 2016.

[227] B. Xu, J. Li, Y. Wong, M. S. Kankanhalli, and Q. Zhao. Interact as you intend: Intention-driven human-object interaction detection. *arXiv preprint arXiv:1808.09796*, 2018.

[228] B. Xu, Y. Wong, J. Li, Q. Zhao, and M. S. Kankanhalli. Learning to detect human-object interactions with knowledge. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[229] S. Yagcioglu, A. Erdem, E. Erdem, and N. Ikizler-Cinbis. Recipeqa: A challenge dataset for multimodal comprehension of cooking recipes. *arXiv preprint arXiv:1809.00812*, 2018.

[230] J. Yang, P. Ren, D. Chen, F. Wen, H. Li, and G. Hua. Neural aggregation network for video face recognition. *arXiv preprint arXiv:1603.05474*, 2016.

[231] S. Yang, P. Luo, C. C. Loy, and X. Tang. Wider face: A face detection benchmark. *arXiv preprint arXiv:1511.06523*, 2015.

[232] S. Yang, Y. Xiong, C. C. Loy, and X. Tang. Face detection through scale-friendly deep convolutional networks. *arXiv preprint arXiv:1706.02863*, 2017.

[233] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola. Stacked Attention Networks for Image Question Answering. *Proceedings - IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, (1):21–29, 2016.

[234] B. Yao and L. Fei-Fei. Grouplet: A structured image representation for recognizing human and object interactions. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 9–16. IEEE, 2010.

[235] B. Yao and L. Fei-Fei. Modeling mutual context of object and human pose in human-object interaction activities. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 17–24. IEEE, 2010.

[236] B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. Guibas, and L. Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1331–1338. IEEE, 2011.

[237] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[238] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[239] L. Yu, E. Park, A. C. Berg, and T. L. Berg. Visual Madlibs: Fill in the blank Description Generation and Question Answering. *Proceedings - IEEE International Conference on Computer Vision, ICCV*, 2015.

[240] R. Yu, X. Chen, V. I. Morariu, and L. S. Davis. The role of context selection in object detection. *arXiv preprint arXiv:1609.02948*, 2016.

[241] H. Zhang, X. Shang, W. Yang, H. Xu, H. Luan, and T.-S. Chua. Online collaborative learning for open-vocabulary visual classifiers. In *CVPR*, pages 2809–2817. IEEE, 2016.

[242] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li. S^3fd: Single shot scale-invariant face detector. *arXiv preprint arXiv:1708.05237*, 2017.

[243] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao. Range loss for deep face recognition with long-tailed training data. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5419–5428. IEEE, 2017.

[244] Y. Zhang, L. Yuan, Y. Guo, Z. He, I.-A. Huang, and H. Lee. Discriminative bimodal networks for visual localization and detection with natural language queries. *arXiv preprint arXiv:1704.03944*, 2017.

[245] Z. Zhang and V. Saligrama. Zero-shot learning via joint latent similarity embedding. In *CVPR*, pages 6034–6042. IEEE, 2016.

[246] Z. Zhang and V. Saligrama. Zero-shot recognition via structured prediction. In *ECCV*, pages 533–548. Springer, 2016.

[247] J. Zhao, T. Wang, M. Yatskar, V. Ordonez, and K.-W. Chang. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. *arXiv preprint arXiv:1707.09457*, 2017.

[248] Y. Zheng, D. K. Pal, and M. Savvides. Ring loss: Convex feature normalization for face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5089–5097, 2018.

[249] P. Zhou and M. Chi. Relation parsing neural network for human-object interaction detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[250] C. Zhu, Y. Zheng, K. Luu, and M. Savvides. Cms-rcnn: contextual multi-scale region-based cnn for unconstrained face detection. *arXiv preprint arXiv:1606.05413*, 2016.

[251] P. Zhu, H. Wang, T. Bolukbasi, and V. Saligrama. Zero-shot detection. *arXiv preprint arXiv:1803.07113*, 2018.

[252] S. Zhu, C. Li, C.-C. Loy, and X. Tang. Unconstrained face alignment via cascaded compositional learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3409–3417, 2016.

[253] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2879–2886, June 2012.

[254] Y. Zhu, O. Groth, M. Bernstein, and L. Fei-Fei. Visual7W: Grounded Question Answering in Images. *Proceedings - IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, nov 2016.

[255] B. Zhuang, L. Liu, C. Shen, and I. Reid. Towards context-aware interaction recognition. *arXiv preprint arXiv:1703.06246*, 2017.

[256] B. Zhuang, Q. Wu, C. Shen, I. Reid, and A. v. d. Hengel. Care about you: towards large-scale human-centric visual relationship detection. *arXiv preprint arXiv:1705.09892*, 2017.

[257] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision*, pages 391–405. Springer, 2014.