VISUAL GENOME
CROWDSOURCED VISUAL KNOWLEDGE REPRESENTATIONS

A THESIS
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTERS OF SCIENCE

Ranjay Krishna
March 2016

Abstract

Despite progress in perceptual tasks such as image classification, computers still perform poorly on cognitive tasks such as image description and question answering. Cognition is core to tasks that involve not just recognizing, but reasoning about our visual world. However, models used to tackle the rich content in images for cognitive tasks are still being trained using the same datasets designed for perceptual tasks. To achieve success at cognitive tasks, models need to understand the interactions and relationships between objects in an image. When asked "What vehicle is the person riding?", computers will need to identify the objects in an image as well as the relationships *riding(man, carriage)* and *pulling(horse, carriage)* in order to answer correctly that "the person is riding a horse-drawn carriage."

My research goal in this thesis is to develop a structured knowledge representation of the visual world and demonstrate its utility. This thesis develops the notion of a scene graph, the representation used for a single image. This representation contains all the recognizable objects in the image along with their attribute modifiers and pairwise relationships. It grounds visual information in a meaning space that allows us to jointly reason about the contents of all images.

In the beginning of this thesis, I present the Visual Genome project. I led the project in collecting the Visual Genome dataset, which maps images to its human generated scene graph. The dataset contains dense annotations of objects, attributes, and relationships within each image to learn cognitive models. The dataset also contains unconstrained image descriptions and visual question answers. My colleagues and I canonicalize the objects, attributes, relationships, and noun phrases in descriptions and questions answer pairs to WordNet synsets. Together, these annotations represent the densest and largest dataset of image descriptions, objects, attributes, relationships, and question answers.

Next, I describe the crowdsourcing techniques developed to collect the Visual Genome dataset. I outline the lessons learned to transfer these strategies to collect other large scale language and vision datasets. This thesis also introduces a new crowdsourcing technique inspired by rapid serial visual processing. It evaluates the technique on a breadth of common labeling tasks such as image verification, word similarity, sentiment analysis and topic classification. Where prior work typically achieves a $0.25\times$ to $1\times$ speedup over fixed majority vote, this approach often achieves an order of magnitude (10×) speedup.

Once the data has been collected, I demonstrate how machine learning models can be built to automatically detect the objects in images and also extract relationships between them. Complementing relationship extraction from images, this thesis also explores how scene graphs can be extracted from sentences alone. Together, we show how semantic image search can be improved by converting natural language queries into scene graphs and then mapping them to images.

Finally, I suggest multiple future applications for the Visual Genome dataset. Overall, this thesis focuses on how a visual knowledge representation allows for a multi-perspective study of an image, from pixel-level information like objects, to relationships that require further inference and cognition.

Acknowledgments

First and foremost, I would like to thank my advisors Fei-Fei Li and Michael Bernstein for fostering my growth as an academic researcher. There high level vision and low level guidance have helped me grow both as a researcher as well as a mentor for other students.

Next, I would like to thank all my co-authors whose work is featured in this thesis (in alphabetical order): Angel Chang, Cewu Lu, Christopher Manning, David A. Shamma, Joshua Kravitz, Justin Johnson, Kenji Hata, Li Fei-Fei, Li Jia-Li, Michael Bernstein, Oliver Groth, Stephanie Chen, Sebastian Schuster, Yannis Kalanditis and Yuke Zhu. Thank you to all the members of both the Stanford Vision Lab and the Stanford HCI Lab for the helpful discussions that contributed to this work.

Thank you to all my research mentors: David A. Shamma and Li Jia-Li for always being available with the perfect solutions to all my challenges. I would also like to thank the students I mentored at Stanford: Asli Kamya, Frederic Ren, Gavin Mai, Kenji Hata, Joshua Kravitz, Michelle Guo, Sherman Leung, Stephanie Chen and Yutian Li.

Thank you to the Brown Intitute of Media Innovation, Stanford Computer Science Department, Yahoo Labs!, Toyota and Adobe for sponsoring parts of this research.

Finally, thank you to me family, friends and loved ones. Thank you to Tanya Dua for always helping me find a balance between work and life. Thank you to my sister for always being a comic relief. Thank you to my Mom and Dad for helping me get to Stanford and encouraging me to follow my interests.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

A holy grail of computer vision is the complete understanding of visual scenes: a model that is able to name and detect objects, describe their attributes, and recognize their relationships and interactions. Understanding scenes would enable important applications such as image search, question answering, and robotic interactions. Much progress has been made in recent years towards this goal, including image classification [40, 178, 217, 116, 231] and object detection [46, 71, 211, 70, 191]. An important contributing factor is the availability of a large amount of data that drives the statistical models that underpin today's advances in computational visual understanding. While the progress is exciting, we are still far from reaching the goal of comprehensive scene understanding. As Figure 1.1 shows, existing models would be able to detect discreet objects in a photo but would not be able to explain their interactions or the relationships between them. Such explanations tend to be *cognitive* in nature, integrating *perceptual* information into conclusions about the relationships between objects in a scene [17, 58]. A cognitive understanding of our visual world thus requires that we complement computers' ability to detect objects with abilities to describe those objects [93] and understand their interactions within a scene [203].

There is an increasing effort to put together the next generation of datasets to serve as training and benchmarking datasets for these deeper, cognitive scene understanding and reasoning tasks, the most notable being MS-COCO [139] and VQA [3]. The MS-COCO dataset consists of 300K real-world photos collected from Flickr. For each image, there is pixel-level segmentation of 91 object classes (when present) and 5 independent, user-generated sentences describing the scene. VQA adds to this a set of 614K question-answer pairs related to the visual contents of each image (see more details in Chapter 2). With this information, MS-COCO and VQA provide a fertile training and testing ground for models aimed at tasks for accurate object detection, segmentation, and summary-level image captioning [108, 150, 106, 242] as well as basic QA [190, 3, 146, 66, 145]. For example,

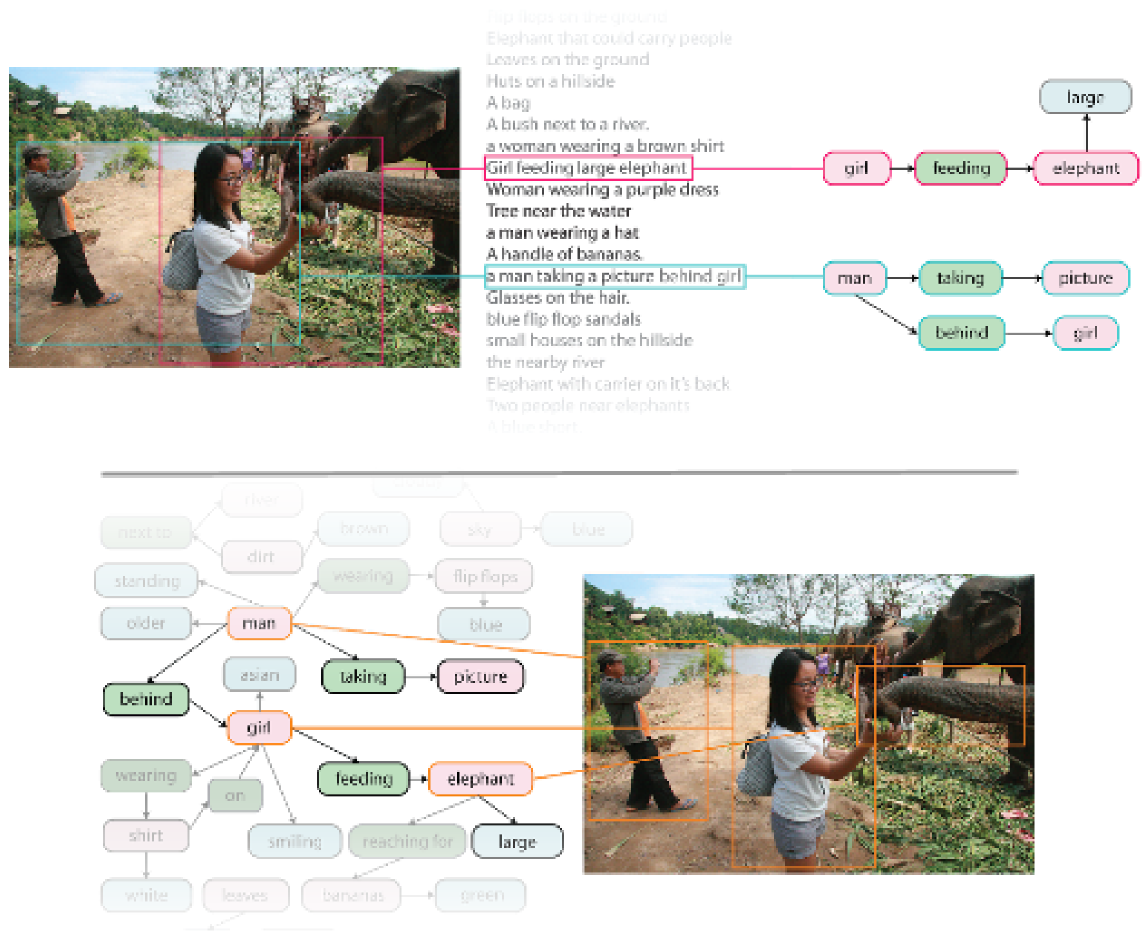


Figure 1.1: An overview of the data needed to move from perceptual awareness to cognitive understanding of images. We present a dataset of images densely annotated with numerous region descriptions, objects, attributes, and relationships. Region descriptions (e.g. "girl feeding large elephant" and "a man taking a picture behind girl") are shown (top). The objects (e.g. `elephant`), attributes (e.g. `large`) and relationships (e.g. `feeding`) are shown (bottom). Our dataset also contains image related question answer pairs (not shown).

a state-of-the-art model [106] provides a description of one MS-COCO image in Figure 1.1 as "two men are standing next to an elephant." But what is missing is the further understanding of where each object is, what each person is doing, what the relationship between the person and elephant is, etc. Without such relationships, these models fail to differentiate this image from other images of people next to elephants.

To understand images thoroughly, we believe three key elements need to be added to existing datasets: a **grounding of visual concepts to language** [108], a more **complete set of descriptions and QAs** for each image based on multiple image regions [100], and a **formalized representation** of the components of an image [80]. In the spirit of mapping out this complete information of the visual world, we introduce the Visual Genome dataset. The first release of the Visual Genome dataset uses 108, 249 images from the intersection of the YFCC100M [235] and MS-COCO [139]. Chapter 2 provides a more detailed description of the dataset. We highlight below the motivation and contributions of the three key elements that set Visual Genome apart from existing datasets.

The Visual Genome dataset regards relationships and attributes as first-class citizens of the annotation space, in addition to the traditional focus on objects. Recognition of relationships and attributes is an important part of the complete understanding of the visual scene, and in many cases, these elements are key to the story of a scene (e.g., the difference between "a dog chasing a man" versus "a man chasing a dog"). The Visual Genome dataset is among the first to provide a detailed labeling of object interactions and attributes, **grounding visual concepts to language**[1].

An image is often a rich scenery that cannot be fully described in one summarizing sentence. The scene in Figure 1.1 contains multiple "stories": "a man taking a photo of elephants," "a woman feeding an elephant," "a river in the background of lush grounds," etc. Existing datasets such as Flickr 30K [262] and MS-COCO [139] focus on high-level descriptions of an image[2]. Instead, for each image in the Visual Genome dataset, we collect more than 42 descriptions for different regions in the image, providing a much denser and **complete set of descriptions of the scene**. In addition, inspired by VQA [3], we also collect an average of 17 question-answer pairs based on the descriptions for each image. Region-based question answers can be used to jointly develop NLP and vision models that can answer questions from either the description or the image, or both of them.

With a set of dense descriptions of an image and the explicit correspondences between visual pixels (i.e. bounding boxes of objects) and textual descriptors (i.e. relationships, attributes), the Visual Genome dataset is poised to be the first image dataset that is capable of providing a structured **formalized representation** of an image, in the form that is widely used in knowledge base representations in NLP [268, 77, 37, 223]. For example, in Figure 1.1, we can formally express the relationship `holding` between the `woman` and `food` as *holding*(*woman*, *food*)). Putting together all the objects and relations in a scene, we can represent each image as a scene graph [100]. The scene graph representation has been shown to improve semantic image retrieval [100, 209] and image captioning [50, 24, 78]. Furthermore, all objects, attributes and relationships in each image in the Visual Genome dataset are canonicalized to its corresponding WordNet [159] ID (called a synset ID). This mapping connects all images in Visual Genome and provides an effective way to consistently query the same concept (object, attribute, or relationship) in the dataset. It can also potentially

[1]The Lotus Hill Dataset [259] also provides a similar annotation of object relationships, see Chapter 1.2.

[2]COCO has multiple sentences generated independently by different users, all focusing on providing an overall, one sentence description of the scene.

help train models that can learn from contextual information from multiple images.

1.1.1 Thesis Outline

In this thesis, we introduce the Visual Genome dataset with the aim of training and benchmarking the next generation of computer models for comprehensive scene understanding. The thesis proceeds as follows: In Chapter 2, we provide a detailed description of each component of the Visual Genome knowledge representation. Chapter 3 discusses the crowdsourcing strategies we deployed in the ongoing effort of collecting this data. Chapter 4 focuses on a particular crowdsourcing technique that we utilized to rapidly collect binary labels by modeling worker latency. Chapter 5 provides a set of experimental results that use Visual Genome as a benchmark; Specifically, this chapter presents experimental results for attribute classification, relationship classification, region description generation and visual question answering. Chapter 6 presents a model for automatically extracting visual relationships from images by leveraging existing language models. Chapter 7 demonstrate the utility of the knowledge representation in semantic image search. The thesis concludes in Chapter 8 where we briefly list potential future directions and applications for Visual Genome.

Further visualizations, API access, dataset navigation and additional information on the Visual Genome dataset can be found online[3].

1.1.2 Previously Published Papers

Most contributions in this thesis have first appeared as various publications. These publications are: [114] (Chapter 1, 2, 3, 5), [113] (Chapter 4), [142] (Chapter 6) and [100, 209] (Chapter 7). My other publications [227] are out of the context of this thesis.

[3]`https://visualgenome.org`

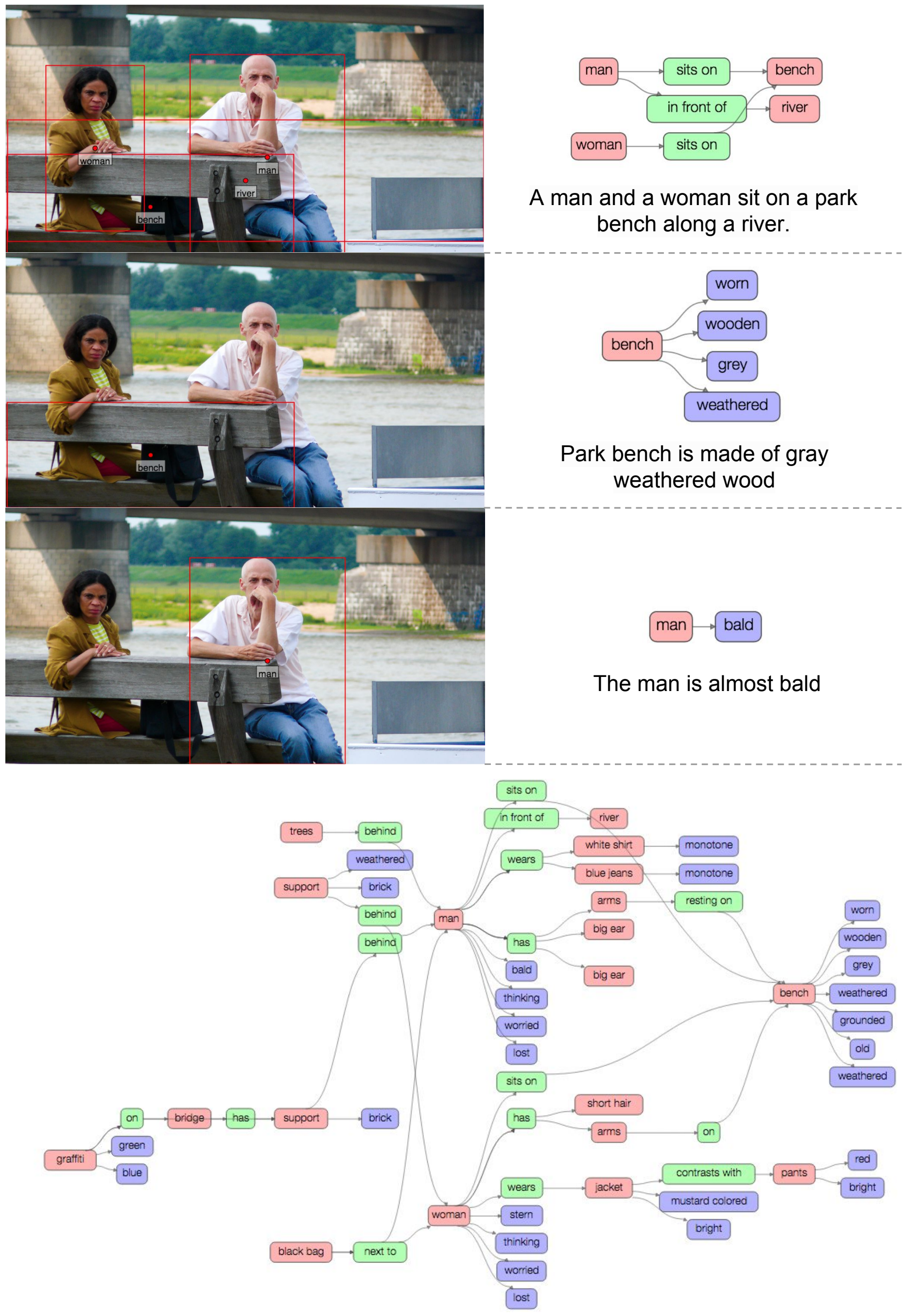


Figure 1.2: An example image from the Visual Genome dataset. We show 3 region descriptions and their corresponding region graphs. We also show the connected scene graph collected by combining all of the image's region graphs. The top region description is "a man and a woman sit on a park bench along a river." It contains the objects: `man`, `woman`, `bench` and `river`. The relationships that connect these objects are: *sits_on*(*man*, *bench*), *in_front_of*(*man*, *river*), and *sits_on*(*woman*, *bench*).

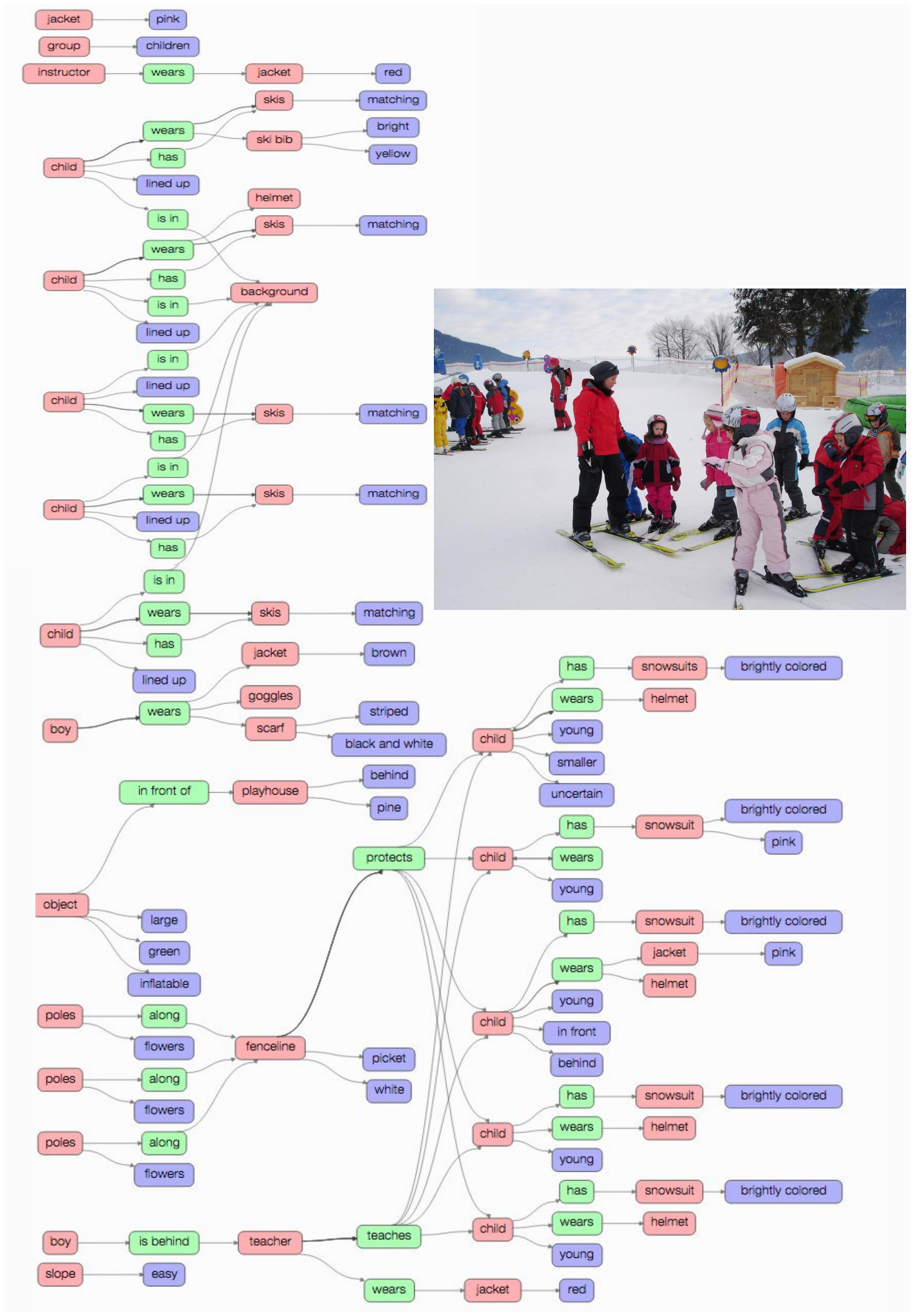


Figure 1.3: An example image from our dataset along with its scene graph representation. The scene graph contains objects (`child`, `instructor`, `helmet`, etc.) that are localized in the image as bounding boxes (not shown). These objects also have attributes: `large`, `green`, `behind`, etc. Finally, objects are connected to each other through relationships: *wears*(*child*, *helmet*), *wears*(*instructor*, *jacket*), etc.

1.2 Background and Related Work

We discuss existing datasets that have been released and used by the vision community for classification and object detection. We also mention work that has improved object and attribute detection models. Then, we explore existing work that has utilized representations similar to our relationships between objects. In addition, we dive into literature related to cognitive tasks like image description, question answering, and knowledge representation.

1.2.1 Datasets

Datasets (Table 1.1) have been growing in size as researchers have begun tackling increasingly complicated problems. *Caltech 101* [53] was one of the first datasets hand-curated for image classification, with 101 object categories and 15-30 of examples per category. One of the biggest criticisms of Caltech 101 was the lack of variability in its examples. *Caltech 256* [75] increased the number of categories to 256, while also addressing some of the shortcomings of Caltech 101. However, it still had only a handful of examples per category, and most of its images contained only a single object. *LabelMe* [201] introduced a dataset with multiple objects per category. They also provided a web interface that experts and novices could use to annotate additional images. This web interface enabled images to be labeled with polygons, helping create datasets for image segmentation. The *Lotus Hill dataset* [259] contains a hierarchical decomposition of objects (vehicles, man-made objects, animals, etc.) along with segmentations. Only a small part of this dataset is freely available. *SUN* [253], just like LabelMe [201] and Lotus Hill [259], was curated for object detection. Pushing the size of datasets even further, 80 *Million Tiny Images* [237] created a significantly larger dataset than its predecessors. It contains tiny (i.e. 32×32 pixels) images that were collected using WordNet [159] synsets as queries. However, because the data in 80 Million Images were not human-verified, they contain numerous errors. *YFCC100M* [235] is another large database of 100 million images that is still largely unexplored. It contains human generated and machine generated tags.

Pascal VOC [46] pushed research from classification to object detection with a dataset containing 20 semantic categories in $11,000$ images. *Imagenet* [40] took WordNet synsets and crowdsourced a large dataset of 14 million images. They started the ILSVRC [197] challenge for a variety of computer vision tasks. ILSVRC and PASCAL provide a test bench for object detection, image classification, object segmentation, person layout, and action classification. *MS-COCO* [139] recently released its dataset, with over $328,000$ images with sentence descriptions and segmentations of 91 object categories. The current largest dataset for QA, *VQA* [3], contains $204,721$ images annotated with one or more question answers. They collected a dataset of $614,163$ freeform questions with 6.1M ground truth answers and provided a baseline approach in answering questions using an image and a textual question as the input.

Visual Genome aims to bridge the gap between all these datasets, collecting not just annotations

	Images	Descriptions per Image	Total Objects	# Object Categories	Objects per Image	# Attributes Categories	Attributes per Image	# Relationship Categories	Relationships per Image	Question Answers
YFCC100M [235]	100,000,000	-	-	-	-	-	-	-	-	-
Tiny Images [237]	80,000,000	-	-	53,464	1	-	-	-	-	-
ImageNet [40]	14,197,122	-	14,197,122	21,841	1	-	-	-	-	-
ILSVRC Det. [197]	476,688	-	534,309	200	2.5	-	-	-	-	-
MS-COCO [194]	328,000	5	27,472	91	-	-	-	-	-	-
Flickr 30K [262]	30,000	5	-	-	-	-	-	-	-	-
Caltech 101 [53]	9,144	-	9,144	102	1	-	-	-	-	-
Caltech 256 [75]	30,608	-	30,608	257	1	-	-	-	-	-
Caltech Pedestrian [44]	250,000	-	350,000	1	1.4	-	-	-	-	-
Pascal Detection [46]	11,530	-	27,450	20	2.38	-	-	-	-	-
Abstract Scenes [272]	10,020	-	58	11	5	-	-	-	-	-
aPascal [50]	12,000	-	-	-	-	64	-	-	-	-
Animal Attributes [125]	30,000	-	-	-	-	1,280	-	-	-	-
SUN Attributes [174]	14,000	-	-	-	-	700	700	-	-	-
Caltech Birds [245]	11,788	-	-	-	-	312	312	-	-	-
COCO Actions [194]	10,000	-	-	-	-	-	-	140	-	-
Visual Phrases [203]	-	-	-	-	-	-	-	17	1	-
Viske [202]	-	-	-	-	-	-	-	6500	-	-
DAQUAR [145]	1,449	-	-	-	-	-	-	-	-	12,468
COCO QA [190]	123,287	-	-	-	-	-	-	-	-	117,684
Baidu [66]	120,360	-	-	-	-	-	-	-	-	250,569
VQA [3]	204,721	-	-	-	-	-	-	-	-	614,163
Visual Genome	108,000	50	4,102,818	76,340	16	15,626	16	47	18	1,773,258

Table 1.1: A comparison of existing datasets with Visual Genome. We show that Visual Genome has an order of magnitude more descriptions and question answers. It also has a more diverse set of object, attribute, and relationship classes. Additionally, Visual Genome contains a higher density of these annotations per image.

for a large number of objects but also scene graphs, region descriptions, and question answer pairs for image regions. Unlike previous datasets, which were collected for a single task like image classification, the Visual Genome dataset was collected to be a general-purpose representation of the visual world, without bias toward a particular task. Our images contain an average of 21 objects, which is almost an order of magnitude more dense than any existing vision dataset. Similarly, we contain an average of 18 attributes and 18 relationships per image. We also have an order of magnitude more unique objects, attributes, and relationships than any other dataset. Finally, we have 1.7 million question answer pairs, also larger than any other dataset for visual question answering.

1.2.2 Image Descriptions

One of the core contributions of Visual Genome is its descriptions for multiple regions in an image. As such, we mention other image description datasets and models in this subsection. Most work related to describing images can be divided into two categories: retrieval of human-generated captions and generation of novel captions. Methods in the first category use similarity metrics between image features from predefined models to retrieve similar sentences [168, 85]. Other methods map both sentences and their images to a common vector space [168] or map them to a space of triples [49]. Among those in the second category, a common theme has been to use recurrent neural networks to produce novel captions [108, 150, 106, 242]. More recently, researchers have also used a visual attention model [254].

One drawback of these approaches is their attention to describing only the most salient aspect of the image. This problem is amplified by datasets like Flickr 30K [262] and MS-COCO [139], whose sentence desriptions tend to focus, somewhat redundantly, on these salient parts. For example, "an elephant is seen wandering around on a sunny day," "a large elephant in a tall grass field," and "a very large elephant standing alone in some brush" are 3 descriptions from the MS-COCO dataset, and all of them focus on the salient elephant in the image and ignore the other regions in the image. Many real-world scenes are complex, with multiple objects and interactions that are best described using multiple descriptions [106, 130]. Our dataset pushes toward a complete understanding of an image by collecting a dataset in which we capture not just scene-level descriptions but also myriad of low-level descriptions, the "grammar" of the scene.

1.2.3 Objects

Object detection is a fundamental task in computer vision, with applications ranging from identification of faces in photo software to identification of other cars by self-driving cars on the road. It involves classifying an object into a semantic category and localizing the object in the image. Visual Genome uses objects as a core component on which each visual scene is built. Early datasets include the face detectio [88] and pedestrian datasets [44]. The PASCAL VOC and ILSVRC's detection dataset [40] pushed research in object detection. But the images in these datasets are iconic and do not capture the settings in which these objects usually co-occur. To remedy this problem, MS-COCO [139] annotated real-world scenes that capture object contexts. However, MS-COCO was unable to describe all the objects in its images, since they annotated only 91 object categories. In the real world, there are many more objects that the ones captured by existing datasets. Visual Genome aims at collecting annotations for all visual elements that occur in images, increasing the number of semantic categories to over 17,000.

1.2.4 Attributes

The inclusion of attributes allows us to describe, compare, and more easily categorize objects. Even if we haven't seen an object before, attributes allow us to infer something about it; for example, "yellow and brown spotted with long neck" likely refers to a giraffe. Initial work in this area involved finding objects with similar features [147] using examplar SVMs. Next, textures were used to study objects [240], while other methods learned to predict colors [55]. Finally, the study of attributes was explicitly demonstrated to lead to improvements in object classification [50]. Attributes were defined to be paths ("has legs"), shapes ("spherical"), or materials ("furry") and could be used to classify new categories of objects. Attributes have also played a large role in improving fine-grained recognition [73] on fine-grained attribute datasets like CUB-2011 [245]. In Visual Genome, we use a generalized formulation [100], but we extend it such that attributes are not image-specific binaries but rather object-specific for each object in a real-world scene. We also extend the types of attributes to include size ("small"), pose ("bent"), state ("transparent"), emotion ("happy"), and many more.

1.2.5 Relationships

Relationship extraction has been a traditional problem in information extraction and in natural language processing. Syntactic features [268, 77], dependency tree methods [37, 18], and deep neural networks [223, 264] have been employed to extract relationships between two entities in a sentence. However, in computer vision, very little work has gone into learning or predicting relationships. Instead, relationships have been implicitly used to improve other vision tasks. Relative layouts between objects have improved scene categorization [94], and 3D spatial geometry between objects has helped object detection [33]. Comparative adjectives and prepositions between pairs of objects have been used to model visual relationships and improved object localization [78].

Relationships have already shown their utility in improving cognitive tasks. A meaning space of relationships has improved the mapping of images to sentences [49]. Relationships in a structured representation with objects have been defined as a graph structure called a *scene graph*, where the nodes are objects with attributes and edges are relationships between objects. This representation can be used to generate indoor images from sentences and also to improve image search [24, 100]. We use a similar scene graph representation of an image that generalizes across all these previous works [100]. Recently, relationships have come into focus again in the form of question answering about associations between objects [202]. These questions ask if a relationship, involving generally two objects, is true, e.g. “do dogs eat ice cream?”. We believe that relationships will be necessary for higher-level cognitive tasks [100, 142], so we collect the largest corpus of them in an attempt to improve tasks by actually understanding relationships between objects.

1.2.6 Question Answering

Visual question answering (QA) has been recently proposed as a proxy task of evaluating a computer vision system’s ability to understand an image beyond object recognition [68, 145]. Several visual QA benchmarks have been proposed in the last few months. The DAQUAR [145] dataset was the first toy-sized QA benchmark built upon indoor scene RGB-D images of NYU Depth v2 [163]. Most new datasets [263, 190, 3, 66] have collected QA pairs on MS-COCO images, either generated automatically by NLP tools [190] or written by human workers [263, 3, 66].

In previous datasets, most questions concentrated on simple recognition-based questions about the salient objects, and answers were often extremely short. For instance, 90% of DAQUAR answers [145] and 87% of VQA answers [3] consist of single-word object names, attributes, and quantities. This shortness limits their diversity and fails to capture the long-tail details of the images. Given the availability of new datasets, an array of visual QA models have been proposed to tackle QA tasks. The proposed models range from SVM classifiers [3] and probabilistic inference [145] to recurrent neural networks [66, 146, 190] and convolutional networks [143]. Visual Genome aims to capture the details of the images with diverse question types and long answers. These questions should cover a wide range of visual tasks from basic perception to complex reasoning. Our QA dataset of 1.7 million QAs is also larger than any currently existing dataset.

1.2.7 Knowledge Representation

A knowledge representation of the visual world is capable of tackling an array of vision tasks, from action recognition to general question answering. However, it is difficult to answer “what is the minimal viable set of knowledge needed to understand about the physical world?” [80]. It was later proposed that there be a certain plurality to concepts and their related axioms [81]. These efforts have grown to model physical processes [62] or to model a series of actions as scripts [206] for stories—both of which are not depicted in a single static image but which play roles in an image’s story. More recently, NELL [11] learns probabilistic horn clauses by extracting information from the web. DeepQA [56] proposes a probabilistic question answering architecture involving over 100 different techniques. Others have used Markov logic networks [269, 164] as their representation to perform statistical inference for knowledge base construction. Our work is most similar to that of those [26, 270, 271, 202] who attempt to learn common-sense relationships from images. Visual Genome scene graphs can also be considered a *dense* knowledge representation for images. It is similar to the format used in knowledge bases in NLP.

Chapter 2

Knowledge Representation and Statistics

2.1 Visual Genome Knowledge Representation

The Visual Genome dataset consists of seven main components: *region descriptions*, *objects*, *attributes*, *relationships*, *region graphs*, *scene graphs*, and *question-answer pairs*. Figure 2.1 shows examples of each component for one image. To enable research on comprehensive understanding of images, we begin by collecting descriptions and question answers. These are raw texts without any restrictions on length or vocabulary. Next, we extract objects, attributes and relationships from our descriptions. Together, objects, attributes and relationships fabricate our scene graphs that represent a formal representation of an image. In this section, we break down Figure 2.1 and explain each of the seven components. In Chapter 3, we will describe in more detail how data from each component is collected through a crowdsourcing platform.

2.1.1 Multiple regions and their descriptions

In a real-world image, one simple summary sentence is often insufficient to describe all the contents of and interactions in an image. Instead, one natural way to extend this might be a collection of descriptions based on different regions of a scene. In Visual Genome, we collect human-generated image region descriptions, with each region localized by a bounding box. In Figure 2.2, we show three examples of region descriptions. Regions are allowed to have a high degree of overlap with each other when the descriptions differ. For example, "yellow fire hydrant" and "woman in shorts is standing behind the man" have very little overlap, while "man jumping over fire hydrant" has a very high overlap with the other two regions. Our dataset contains on average a total of 42 region descriptions per image. Each description is a phrase ranging from 1 to 16 words in length describing that region.

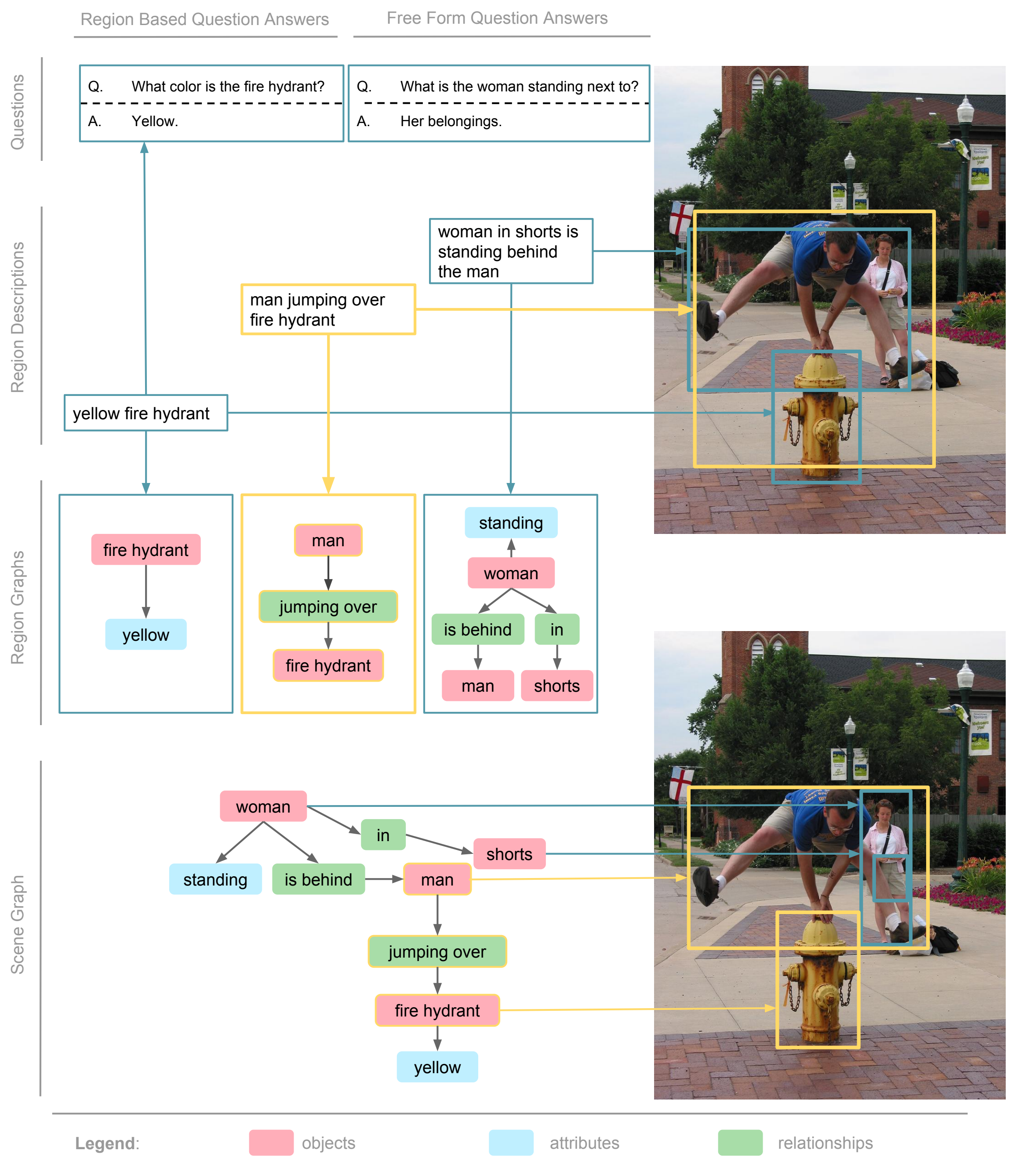


Figure 2.1: A representation of the Visual Genome dataset. Each image contains region descriptions that describe a localized portion of the image. We collect two types of question answer pairs (QAs): freeform QAs and region-based QAs. Each region is converted to a region graph representation of objects, attributes, and pairwise relationships. Finally, each of these region graphs are combined to form a scene graph with all the objects grounded to the image. *Best viewed in color*

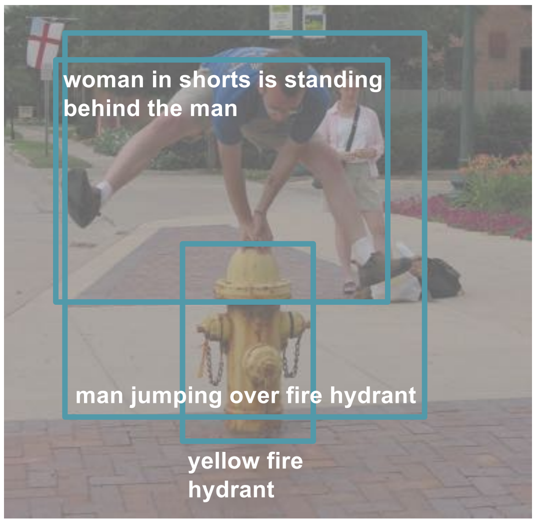

woman in shorts is standing behind the man
man jumping over fire hydrant
yellow fire hydrant

Figure 2.2: To describe all the contents of and interactions in an image, the Visual Genome dataset includes multiple human-generated image regions descriptions, with each region localized by a bounding box. Here, we show three regions descriptions: "man jumping over a fire hydrant," "yellow fire hydrant," and "woman in shorts is standing beghind the man."

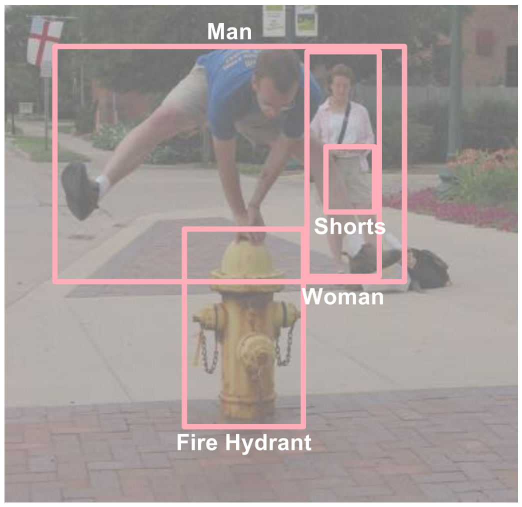


Figure 2.3: From all of the region descriptions, we extract all objects mentioned. For example, from the region description "man jumping over a fire hydrant," we extract `man` and `fire hydrant`.

2.1.2 Multiple objects and their bounding boxes

Each image in our dataset consists of an avarege of 21 objects, each delineated by a tight bounding box (Figure 2.3). Furthermore, each object is canonicalized to a synset ID in WordNet [159]. For example, `man` and `person` would get mapped to `man.n.03 (the generic use of the`

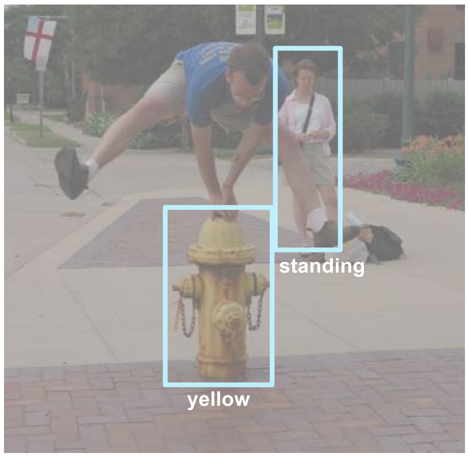


Figure 2.4: Some descriptions also provide attributes for objects. For example, the region description "yellow fire hydrant" adds that the `fire hydrant` is `yellow`. Here we show two attributes: `yellow` and `standing`.

`word to refer to any human being)`. Similarly, `person` gets mapped to `person.n.01` `(a human being)`. Afterwards, these two concepts can be joined to `person.n.01` since this is a hypernym of `man.n.03`. This is an important standardization step to avoid multiple names for one object (e.g. man, person, human), and to connect information across images.

2.1.3 A set of attributes

Each image in Visual Genome has an average of 16 attributes. Objects can have zero or more attributes associated with them. Attributes can be color (`yellow`), states (`standing`), etc. (Figure 2.4). Just like we extract objects from region descriptions, we also extract the attributes attached to these objects. In Figure 2.4, from the phrase "yellow fire hydrant," we extract the attribute `yellow` for the `fire hydrant`. As with objects, we canonicalize all attributes to WordNet [159]; for example, `yellow` is mapped to `yellow.s.01 (of the color intermediate between green and orange in the color spectrum; of something resembling the color of an egg yolk)`.

2.1.4 A set of relationships

Relationships connect two objects together. These relationships can be actions (`jumping over`), spatial (`is behind`), verbs (`wear`), prepositions (`with`), comparative (`taller than`), or prepositional phrases (`drive on`). For example, from the region description "man jumping over fire hydrant," we extract the relationship `jumping over` between the objects `man` and `fire hydrant` (Figure 2.5). These relationships are directed from one object, called the subject, to another, called

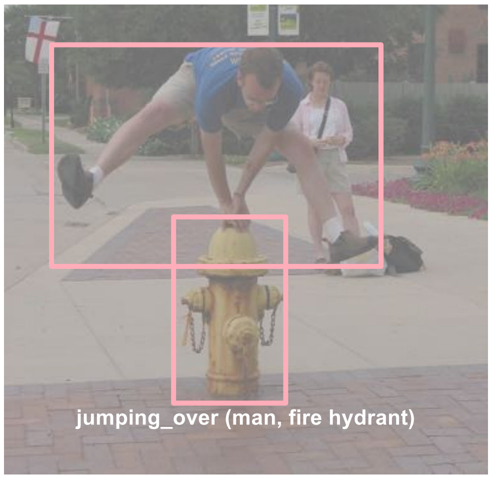


Figure 2.5: Our dataset also captures the relationships and interactions between objects in our images. In this example, we show the relationship `jumping over` between the objects `man` and `fire hydrant`.

the object. In this case, the subject is the `man`, who is performing the relationship `jumping over` on the object `fire hydrant`. Each relationship is canonicalized to a WordNet [159] synset ID; i.e. `jumping` is canonicalized to `jump.a.1 (move forward by leaps and bounds)`. On average, each image in our dataset contains 18 relationships.

2.1.5 A set of region graphs

Combining the objects, attributes, and relationships extracted from region descriptions, we create a directed graph representation for each of the 42 regions. Examples of region graphs are shown in Figure 2.1. Each region graph is a structured representation of a part of the image. The nodes in the graph represent objects, attributes, and relationships. Objects are linked to their respective attributes while relationships link one object to another. The links connecting two objects in Figure 2.1 point from the subject to the relationship and from the relationship to the other object.

2.1.6 One scene graph

While region graphs are localized representations of an image, we also combine them into a single scene graph representing the entire image (Figure 1.3). The scene graph is the *union* of all region graphs and contains all objects, attributes, and relationships from each region description. By doing so, we are able to combine multiple levels of scene information in a more coherent way. For example in Figure 2.1, the leftmost region description tells us that the "fire hydrant is yellow," while the middle region description tells us that the "man is jumping over the fire hydrant." Together, the two descriptions tell us that the "man is jumping over a yellow fire hydrant."

2.1.7 A set of question answer pairs

We have two types of QA pairs associated with each image in our dataset: *freeform QAs*, based on the entire image, and *region-based QAs*, based on selected regions of the image. We collect 6 different types of questions per image: `what`, `where`, `how`, `when`, `who`, and `why`. In Figure 2.1, "Q. What is the woman standing next to?; A. Her belongings" is a freeform QA. Each image has at least one question of each type listed above. Region-based QAs are collected by prompting workers with region descriptions. For example, we use the region "yellow fire hydrant" to collect the region-based QA: "Q. What color is the fire hydrant?; A. Yellow." Region based QAs allow us to independently study methods that use NLP and vision priors to answer questions.

2.2 Dataset Statistics and Analysis

In this section, we provide statistical insights and analysis for each component of Visual Genome. Specifically, we examine the distribution of *images* (Chapter 2.2.1) and the collected data for *region descriptions* (Chapter 2.2) and *questions and answers* (Chapter 2.2.7). We analyze *region graphs* and *scene graphs* together in one section (Chapter 2.2.6), but we also break up these graph structures into their three constituent parts—*objects* (Chapter 2.2.3), *attributes* (Chapter 2.2.4), and *relationships* (Chapter 2.2.5)—and study each part individually. Finally, we describe our canonicalization pipeline and results (Chapter 2.2.8).

2.2.1 Image Selection

The Visual Genome dataset consists of all $108,249$ images from the intersection of MS-COCO's [139] $328,000$ images and YFCC's [235] 100 million images. These images are real-world, non-iconic images that were uploaded onto Flickr by users. The images range from as small as 72 pixels wide to as large as 1280 pixels wide, with an average width of 500 pixels. We collected the WordNet synsets into which our $108,249$ images can be categorized using the same method as ImageNet [40]. Visual Genome images cover 972 synsets. Figure 2.6 shows the top synsets to which our images belong. "ski" is the most common synset, with 2612 images; it is followed by "ballplayer" and "racket," with all three synsets referring to images of people playing sports. Our dataset is somewhat biased towards images of people, as Figure 2.6 shows; however, they are quite diverse overall, as the top 25 synsets each have over 800 images, while the top 50 synsets each have over 500 examples.

2.2.2 Region Description Statistics

One of the primary components of Visual Genome is its region descriptions. Every image includes an average of 42 regions with a bounding box and a descriptive phrase. Figure 2.7 shows an example image from our dataset with its 50 region descriptions. We display bounding boxes for only 6 out of

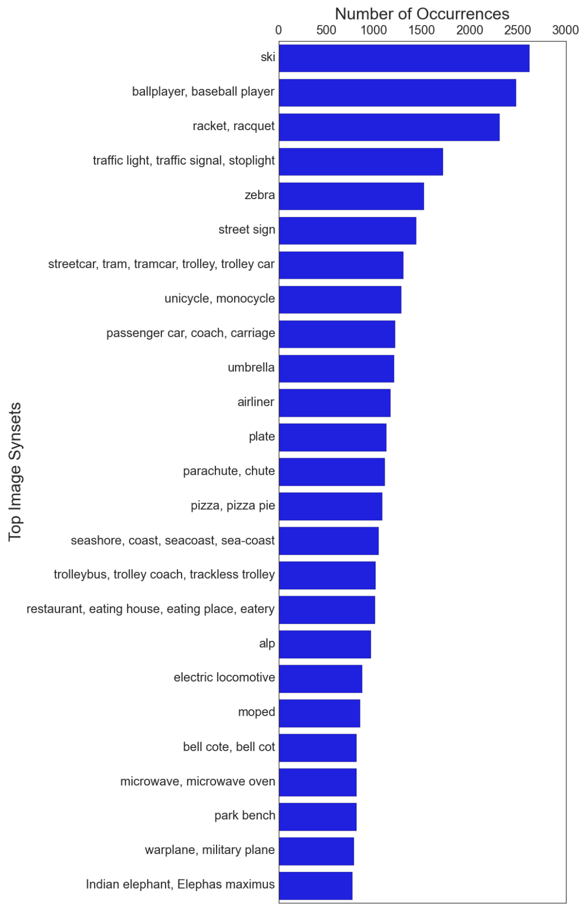
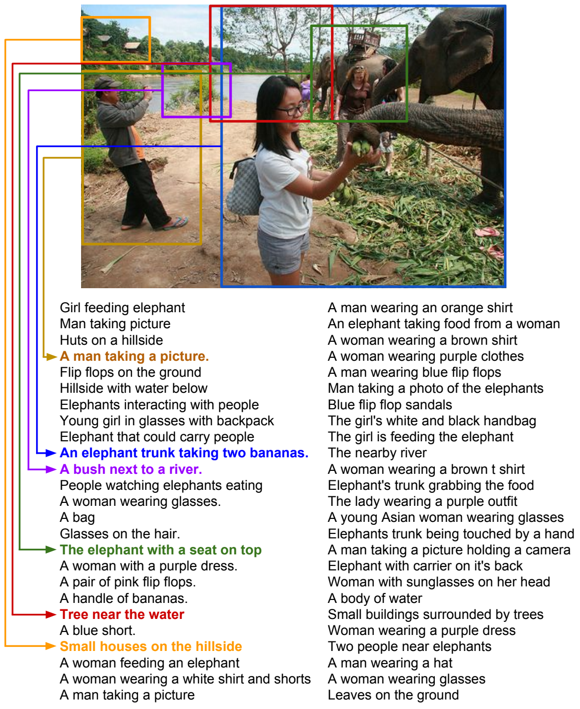


Figure 2.6: A distribution of the top 25 image synsets in the Visual Genome dataset. A variety of synsets are well represented in the dataset, with the top 25 synsets having at least 800 example images each.

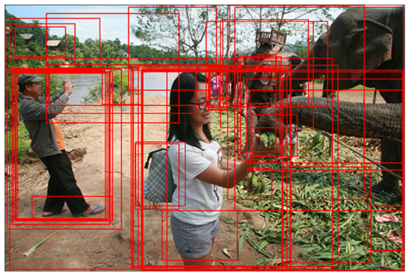
the 50 descriptions in the figure to avoid clutter. These descriptions tend to be highly diverse and can focus on a single object, like in "A bag," or on multiple objects, like in "Man taking a photo of the elephants." They encompass the most salient parts of the image, as in "An elephant taking food from a woman," while also capturing the background, as in "Small buildings surrounded by trees."

MS-COCO [139] dataset is good at generating variations on a single scene-level descriptor. Consider three sentences from MS-COCO dataset on a similar image: "there is a person petting a very large elephant," "a person touching an elephant in front of a wall," and "a man in white shirt petting the cheek of an elephant." These three sentences are single scene-level descriptions. In comparison, Visual Genome descriptions emphasize different regions in the image and thus are less semantically similar. To ensure diversity in the descriptions, we use BLEU score [171] thresholds between new descriptions and all previously written descriptions. More information about crowdsourcing can be



Girl feeding elephant
Man taking picture
Huts on a hillside
A man taking a picture.
Flip flops on the ground
Hillside with water below
Elephants interacting with people
Young girl in glasses with backpack
Elephant that could carry people
An elephant trunk taking two bananas.
A bush next to a river.
People watching elephants eating
A woman wearing glasses.
A bag
Glasses on the hair.
The elephant with a seat on top
A woman with a purple dress.
A pair of pink flip flops.
A handle of bananas.
Tree near the water
A blue short.
Small houses on the hillside
A woman feeding an elephant
A woman wearing a white shirt and shorts
A man taking a picture
A man wearing an orange shirt
An elephant taking food from a woman
A woman wearing a brown shirt
A woman wearing purple clothes
A man wearing blue flip flops
Man taking a photo of the elephants
Blue flip flop sandals
The girl's white and black handbag
The girl is feeding the elephant
The nearby river
A woman wearing a brown t shirt
Elephant's trunk grabbing the food
The lady wearing a purple outfit
A young Asian woman wearing glasses
Elephants trunk being touched by a hand
A man taking a picture holding a camera
Elephant with carrier on it's back
Woman with sunglasses on her head
A body of water
Small buildings surrounded by trees
Woman wearing a purple dress
Two people near elephants
A man wearing a hat
A woman wearing glasses
Leaves on the ground

(a)

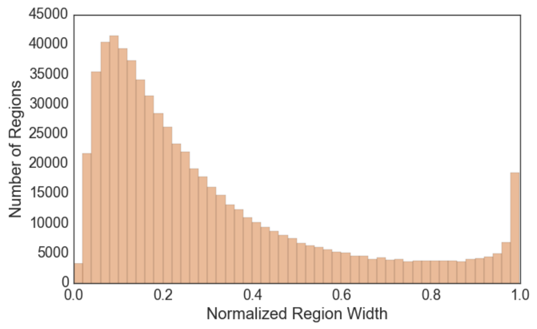


(b)

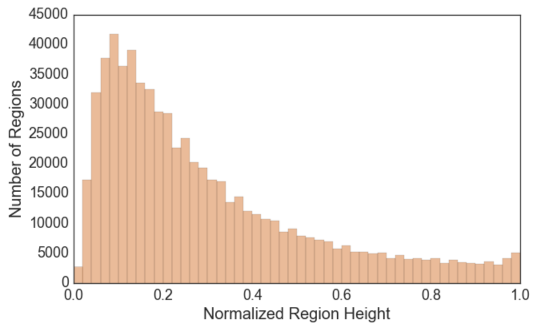
Figure 2.7: (a) An example image from the dataset with its region descriptions. We only display localizations for 6 of the 42 descriptions to avoid clutter; all 50 descriptions do have corresponding bounding boxes. (b) All 42 region bounding boxes visualized on the image.

found in Chapter 3.

Region descriptions must be specific enough in an image to describe individual objects, like in the description "A bag," but they must also be general enough to describe high-level concepts in an image, like "An man being chased by a bear." Qualitatively, we note that regions that cover large portions of the image tend to be general descriptions of an image, while regions that cover only a small fraction of the image tend to be more specific. In Figure 2.8 (a), we show the distribution of regions over the width of the region normalized by the width of the image. We see that the majority of our regions tend to be around 10% to 15% of the image width. We also note that there are a large number of regions covering 100% of the image width. These regions usually include elements like "sky," "ocean," "snow," "mountains," etc. that cannot be bounded and thus span the entire image width. In Figure 2.8 (b), we show a similar distribution over the normalized height of the region. We see a similar overall pattern, as most of our regions tend to be very specific descriptions of about 10% to 15% of the image height. Unlike the distribution over width, however, we do not see a increase in the number of regions that span the entire height of the image, as there are no common visual equivalents that span images vertically. Out of all the descriptions gathered, only



(a)



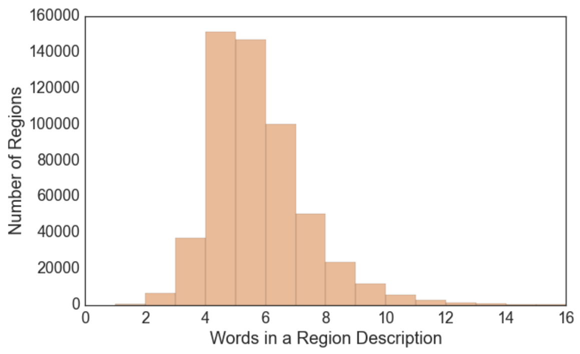
(b)

Figure 2.8: (a) A distribution of the width of the bounding box of a region description normalized by the image width. (b) A distribution of the height of the bounding box of a region description normalized by the image height.

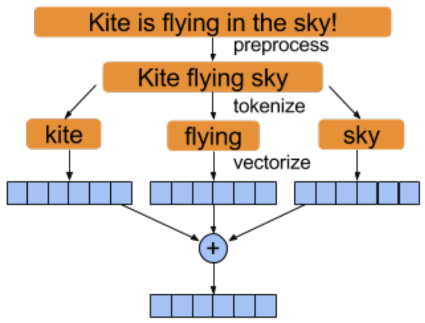
one or two of them tend to be global scene descriptions that are similar to MS-COCO [139].

After examining the distribution of the size of the regions described, it is also valuable to look at the semantic information captured by these descriptions. In Figure 2.9 (a), we show the distribution of the length (word count) of these region descriptions. The average word count for a description is 5 words, with a minimum of 1 word and a maximum of 12 words. In Figure 2.10 (a), we plot the most common phrases occurring in our region descriptions, with stop words removed. Common visual elements like "green grass," "tree [in] distance," and "blue sky" occur much more often than other, more nuanced elements like "fresh strawberry." We also study descriptions with finer precision in Figure 2.10 (b), where we plot the most common words used in descriptions. Again, we eliminate stop words from our study. Colors like "white" and "black" are the most frequently used words to describe visual concepts; we conduct a similar study on other captioning datasets including MS-COCO [139] and Flickr 30K [262] and find a similar distribution with colors occurring most frequently. Besides colors, we also see frequent occurrences of common objects like "man," "tree," and "sign" and of universal visual elements like "sky."

Semantic diversity. We also study the actual semantic contents of the descriptions. We use an unsupervised approach to analyze the semantics of these descriptions. Specifically, we use word2vec [158] to convert each word in a description to a 300-dimensional vector. Next, we remove stop words and average the remaining words to get a vector representation of the whole region description. This pipeline is outlined in Figure 2.9 (b). We use hierarchical agglomerative clustering on vector representations of each region description and find 71 semantic and syntactic groupings or "clusters." Figure 2.11 (a) shows four such example clusters. One cluster contains all descriptions related to tennis, like "A man swings the racquet" and "White lines on the ground of the tennis court," while another cluster contains descriptions related to numbers, like "Three dogs on the street" and



(a)

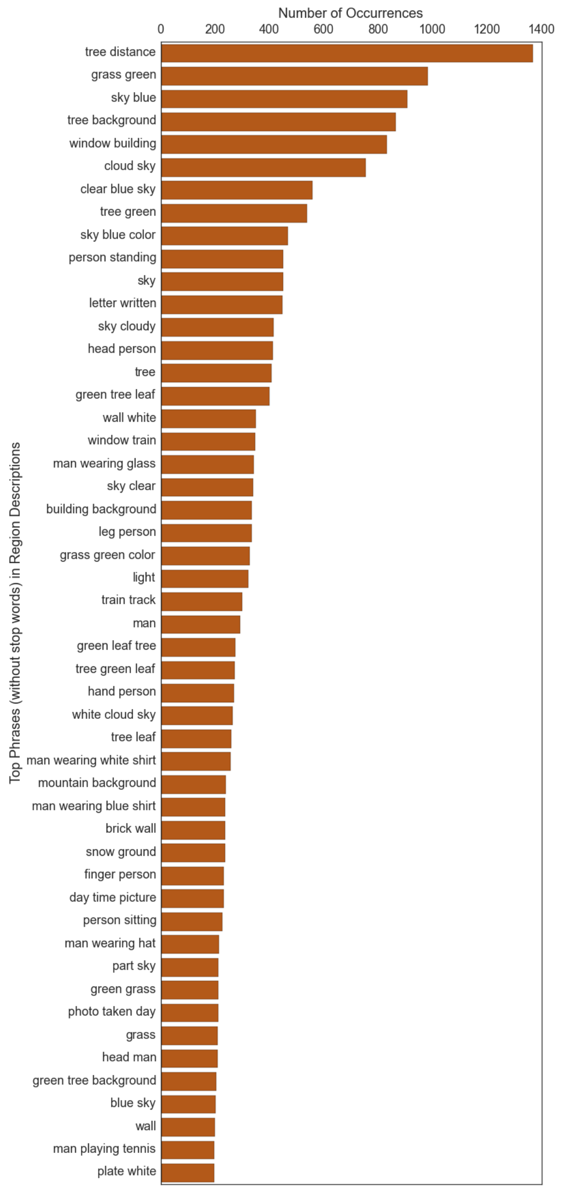


(b)

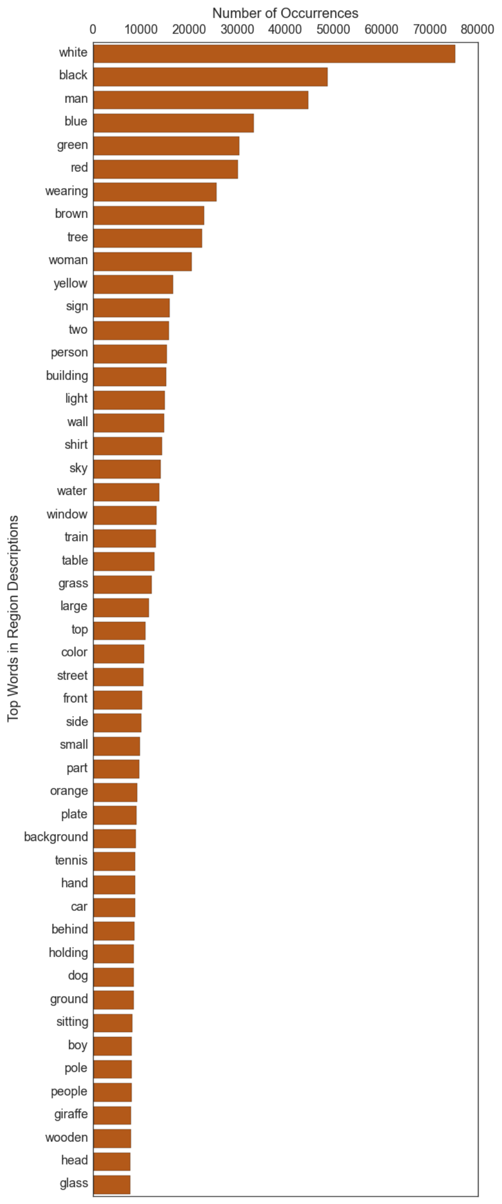
Figure 2.9: (a) A distribution of the number of words in a region description. The average number of words in a region description is 5, with shortest descriptions of 1 word and longest descriptions of 16 words. (b) The process used to convert a region description into a 300-dimensional vectorized representation.

"Two people inside the tent." To quantitatively measure the diversity of Visual Genome's region descriptions, we calculate the number of clusters represented in a single image's region descriptions. We show the distribution of the variety of descriptions for an image in Figure 2.11 (b). We find that on average, each image contains descriptions from 17 different clusters. The image with the least diverse descriptions contains descriptions from 4 clusters, while the image with the most diverse descriptions contains descriptions from 26 clusters.

Finally, we also compare the descriptions in Visual Genome to the captions in MS-COCO. First we aggregate all Visual Genome and MS-COCO descriptions and remove all stop words. After removing stop words, the descriptions from both datasets are roughly the same length. We conduct a similar study, in which we vectorize the descriptions for each image and calculate each dataset's cluster diversity per image. We find that on average, 2 clusters are represented in the captions for each image in MS-COCO, with very few images in which 5 clusters are represented. Because each image in MS-COCO only contains 5 captions, it is not a fair comparison to compare the number of clusters represented in all the region descriptions in the Visual Genome dataset. We thus randomly sample 5 Visual Genome region descriptions per image and calculate the number of clusters in an image. We find that Visual Genome descriptions come from 4 or 5 clusters. We show our comparison results in Figure 2.11 (c). The difference between the semantic diversity between the two datasets is statistically significant ($t = -240$, $p < 0.01$).

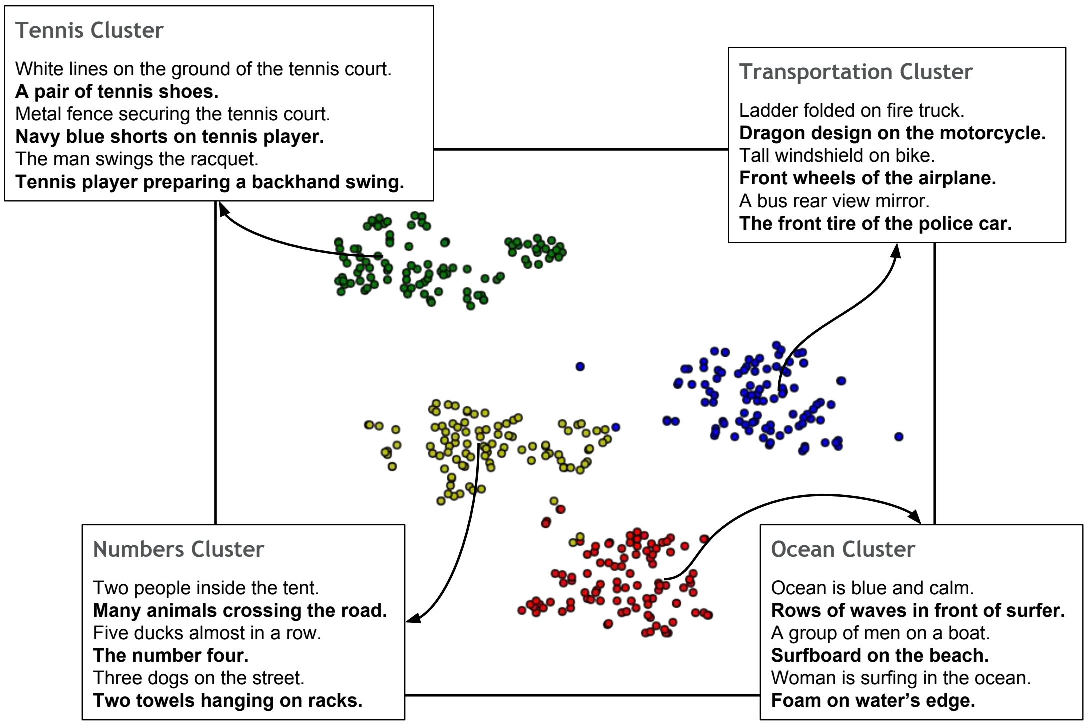


(a)

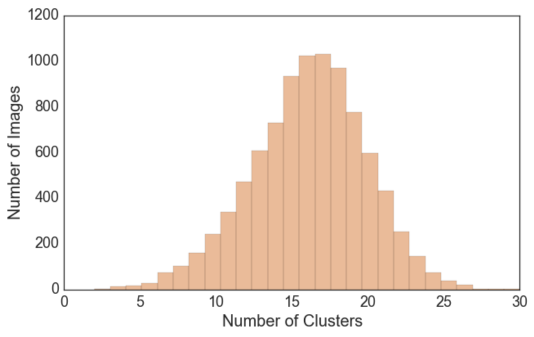


(b)

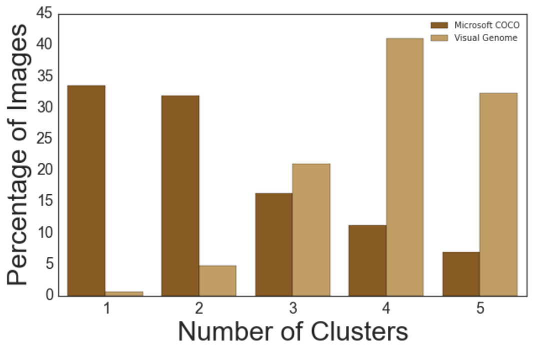
Figure 2.10: (a) A plot of the most common visual concepts or phrases that occur in region descriptions. The most common phrases refer to universal visual concepts like "blue sky," "green grass," etc. (b) A plot of the most frequently used words in region descriptions. Colors occur the most frequently, followed by common objects like "man" and "dog" and universal visual concepts like "sky."


Tennis Cluster
White lines on the ground of the tennis court.
A pair of tennis shoes.
Metal fence securing the tennis court.
Navy blue shorts on tennis player.
The man swings the racquet.
Tennis player preparing a backhand swing.
Transportation Cluster
Ladder folded on fire truck.
Dragon design on the motorcycle.
Tall windshield on bike.
Front wheels of the airplane.
A bus rear view mirror.
The front tire of the police car.
Numbers Cluster
Two people inside the tent.
Many animals crossing the road.
Five ducks almost in a row.
The number four.
Three dogs on the street.
Two towels hanging on racks.
Ocean Cluster
Ocean is blue and calm.
Rows of waves in front of surfer.
A group of men on a boat.
Surfboard on the beach.
Woman is surfing in the ocean.
Foam on water's edge.

(a)

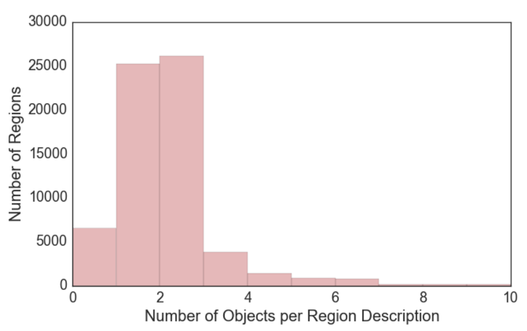


(b)

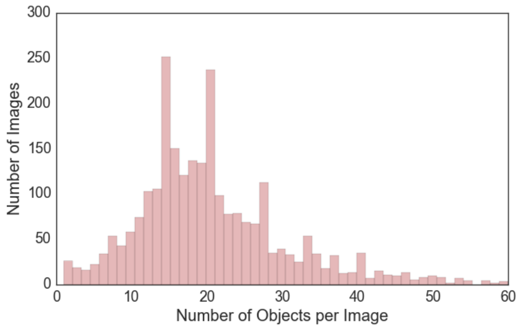


(c)

Figure 2.11: (a) Example illustration showing four clusters of region descriptions and their overall themes. Other clusters not shown due to limited space. (b) Distribution of images over number of clusters represented in each image's region descriptions. (c) We take Visual Genome with 5 random descriptions taken from each image and MS-COCO dataset with all 5 sentence descriptions per image and compare how many clusters are represented in the descriptions. We show that Visual Genome's descriptions are more varied for a given image, with an average of 4 clusters per image, while MS-COCO's images have an average of 3 clusters per image.



(a)



(b)

Figure 2.12: (a) Distribution of the number of objects per region. Most regions have between 0 and 2 objects. (b) Distribution of the number of objects per image. Most images contain between 15 and 20 objects.

	Visual Genome	ILSVRC Det. [197]	MS-COCO [139]	Caltech256 [75]	PASCAL Det. [46]	Abstract Scenes [272]
Images	108,249	476,688	328,000	30,608	11,530	10,020
Total Objects	255,718	534,309	2,500,000	30,608	27,450	58
Total Cat.	18,136	200	91	257	20	11
Objects/Cat.	14.10	2671.50	27472.50	119	1372.50	5.27

Table 2.1: Comparison of Visual Genome objects and categories to related datasets.

2.2.3 Object Statistics

In comparison to related datasets, Visual Genome fares well in terms of object density and diversity. Visual Genome contains approximately 21 objects per image, exceeding ImageNet [40], PASCAL [46], MS-COCO [139], and other datasets by large margins. As shown in Figure 2.13, there are more object categories represented in Visual Genome than in any other dataset. This comparison is especially pertinent with regards to Microsoft MS-COCO [139], which uses the same images as Visual Genome. The lower count of objects per category is a result of our higher number of categories. For a fairer comparison with ILSVRC 2014 Detection [197], Visual Genome has about 2239 objects per category when only the top 200 categories are considered, which is comparable to ILSVRC's 2671.5 objects per category. For a fairer comparison with MS-COCO, Visual Genome has about 3768 objects per category when only the top 91 categories are considered. This is comparable to MS-COCO's [139] when we consider just the $108,249$ MS-COCO images in Visual Genome.

Objects in Visual Genome come from a variety of categories. As shown in Figure 2.14 (b), objects related to WordNet categories such as humans, animals, sports, and scenery are most common; this is consistent with the general bias in image subject matter in our dataset. Common objects like

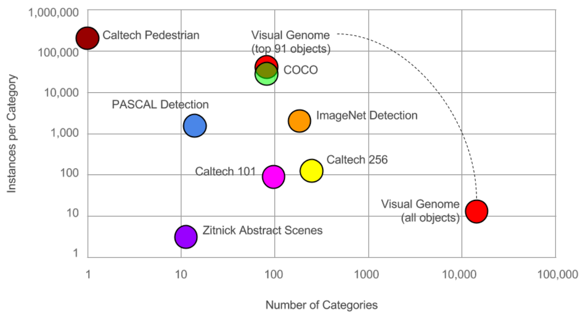
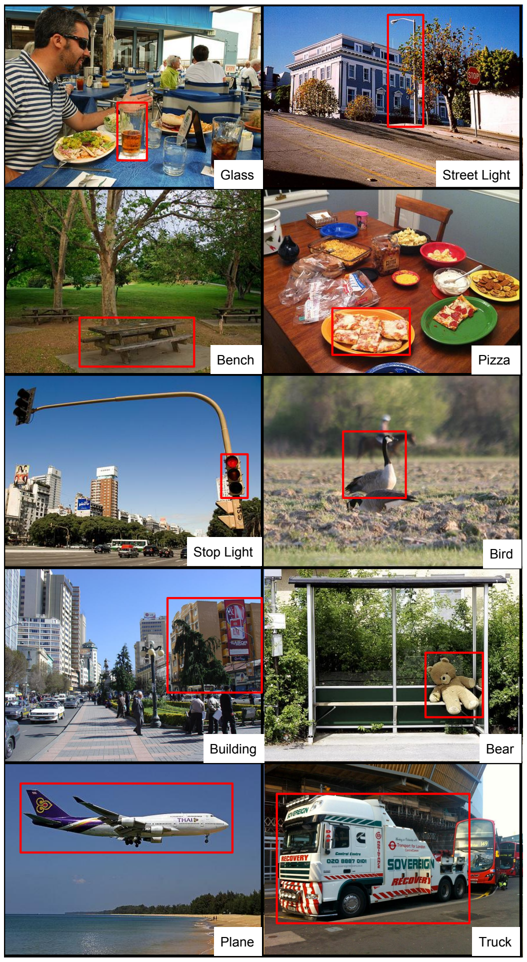


Figure 2.13: Comparison of object diversity between various datasets. Visual Genome far surpasses other datasets in terms of number of object categories.

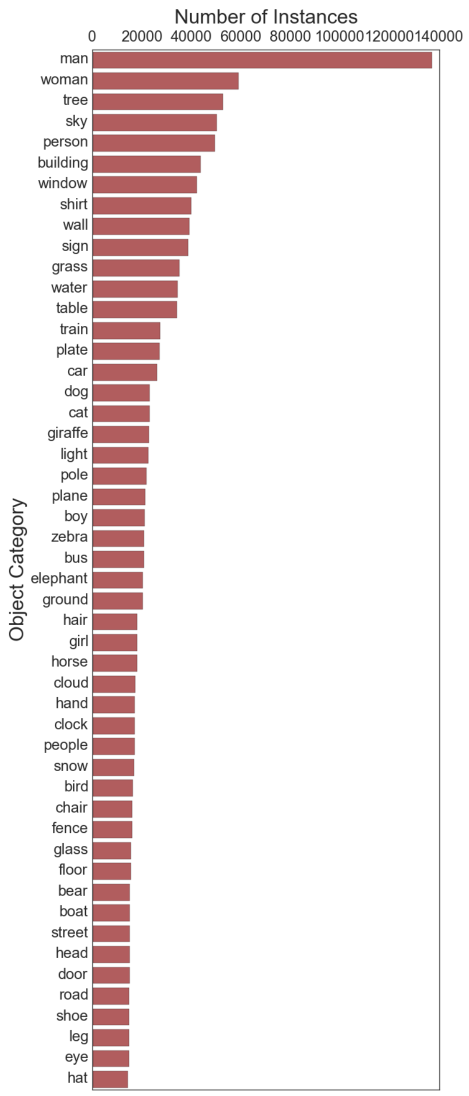
`man`, `person`, and `woman` occur especially frequently with occurrences of 24K, 17K, and 11K. Other objects that also occur in MS-COCO [139] are also well represented with around 5000 instances on average. Figure 2.14 (a) shows some examples of objects in images. Objects in Visual Genome span a diverse set of Wordnet categories like food, animals, and man-made structures.

It is important to look not only at what types of objects we have but also at the distribution of objects in images and regions. Figure 2.12 (a) shows, as expected, that we have between 0 and 2 objects in each region on average. It is possible for regions to contain no objects if their descriptions refer to no explicit objects in the image. For example, a region described as “it is dark outside” has no objects to extract. Regions with only one object generally have descriptions that focus on the attributes of a single object. On the other hand, regions with two or more objects generally have descriptions that contain both attributes of specific objects and relationships between pairs of objects.

As shown in Figure 2.12 (b), each image contains on average around 21 unique objects. Few images have a low number of objects, which we expect since images usually capture more than a few objects. Moreover, few images have an extremely high number of objects (e.g. over 40).

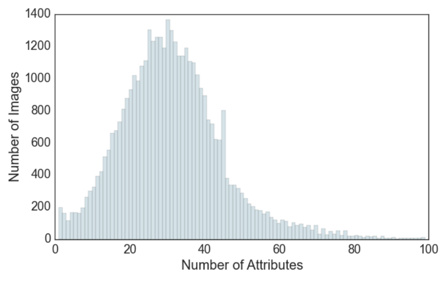


(a)

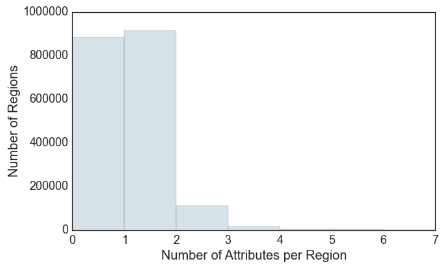


(b)

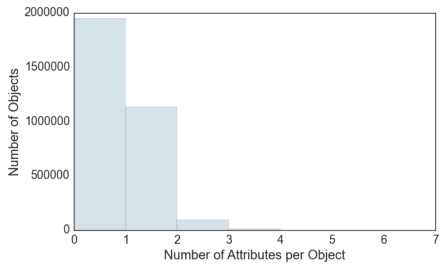
Figure 2.14: (a) Examples of objects in Visual Genome. Each object is localized in its image with a tightly drawn bounding box. (b) Plot of the most frequently occurring objects in images. People are the most frequently occurring objects in our dataset, followed by common objects and visual elements like `building`, `shirt`, and `sky`.



(a)



(b)



(c)

Figure 2.15: Distribution of the number of attributes (a) per image, (b) per region description, (c) per object.

2.2.4 Attribute Statistics

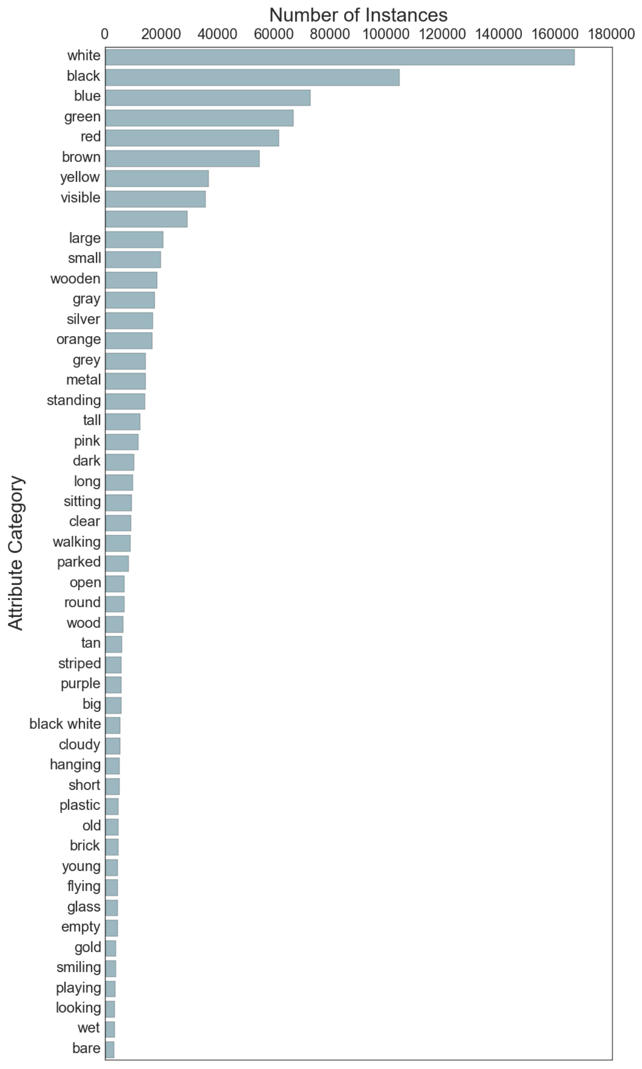
Attributes allow for detailed description and disambiguation of objects in our dataset. About 45% of objects in Visual Genome are annotated with at least one attribute; our dataset contains 1.6 million total attributes with 13,041 unique attributes. Attributes include colors (`green`), sizes (`tall`), continuous action verbs (`standing`), materials (`plastic`), etc. Each attribute in our scene graphs belongs to one object, while each object can have multiple attributes. We denote attributes as `attribute(object)`.

On average, each image in Visual Genome contains 21 attributes, as shown in Figure 2.15. Each region contains on average 1 attribute, though about 42% of regions contain no attribute at all; this is primarily because many regions are relationship-focused. Figure 2.16 (a) shows the distribution of the most common attributes in our dataset. Colors (e.g. `white`, `green`) are by far the most frequent attributes. Also common are sizes (e.g. `large`) and materials (e.g. `wooden`). Figure 2.16 (b) shows the distribution of attributes describing people (e.g. `man`, `girls`, and `person`). The most common attributes describing people are intransitive verbs describing their states of motion (e.g.`standing` and `walking`). Certain sports (`skiing`, `surfboarding`) are overrepresented due to a bias towards these sports in our images.

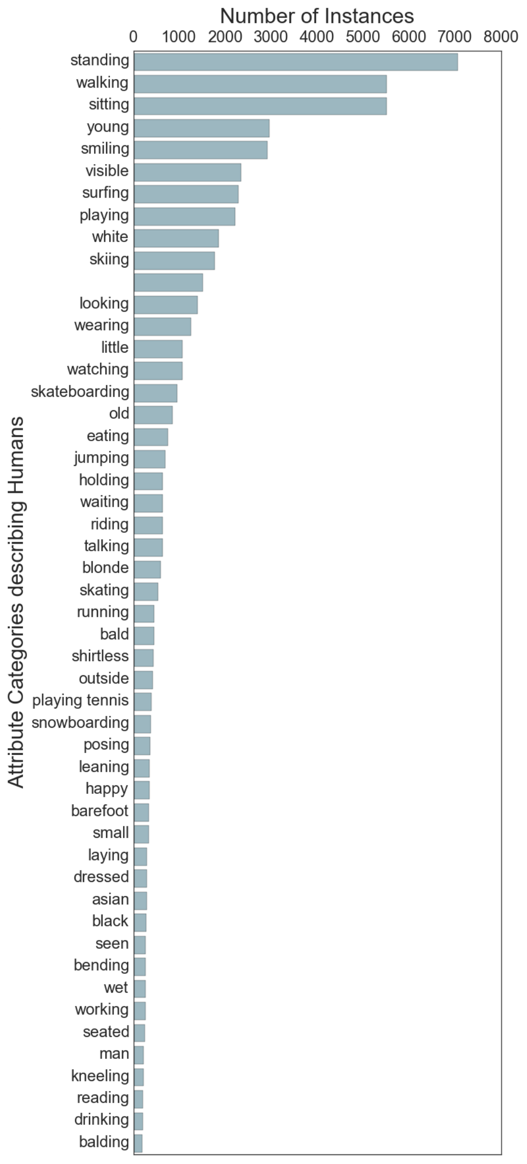
Attribute Graphs. We also qualitatively analyze the attributes in our dataset by constructing co-occurrence graphs, in which nodes are unique attributes and edges connect those attributes that describe the same object. For example, if an image contained a "large black dog" (`large(dog)`, `black(dog)`) and another image contained a "large yellow cat" (`large(cat)`, `yellow(cat)`), its attributes would form an incomplete graph with edges (`large`, `black`) and (`large`, `yellow`). We create two such graphs: one for both the total set of attributes and a second where we consider only objects that refer to people. A subgraph of the 16 most frequently connected (co-occurring) person-related attributes is shown in Figure 2.17 (a).

Cliques in these graphs represent groups of attributes in which at least one co-occurrence exists for each pair of attributes in that group. In the previous example, if a third image contained a "black and yellow taxi" (`black(taxi)`, `yellow(taxi)`), the resulting third edge would create a clique between the attributes `black`, `large`, and `yellow`. When calculated across the entire Visual Genome dataset, these cliques provide insight into commonly perceived traits of different types of objects. Figure 2.17 (b) is a selected representation of three example cliques and their overlaps. From just a clique of attributes, we can predict what types of objects are usually referenced. In Figure 2.17 (b), we see that these cliques describe an animal (left), water body (top right), and human hair (bottom right).

Other cliques (not shown) can also uniquely identify objects. In our set, one clique contains `athletic`, `young`, `fit`, `skateboarding`, `focused`, `teenager`, `male`, `skinny`, and `happy`, capturing some of the common traits of `skateboarders` in our set. Another such clique has `shiny`, `small`, `metal`, `silver`, `rusty`, `parked`, and `empty`, most likely describing a subset of cars. From these cliques, we can thus infer distinct objects and object types based solely on their attributes, potentially allowing for highly specific object identification based on selected characteristics.

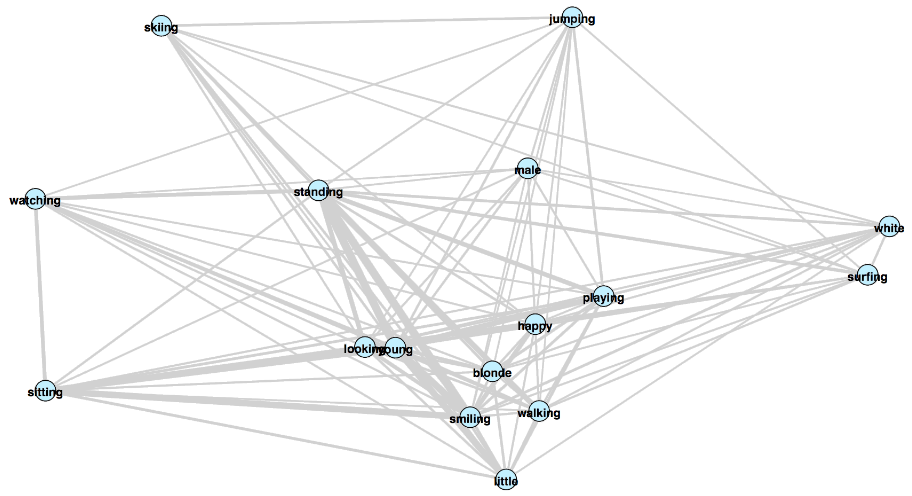


(a)

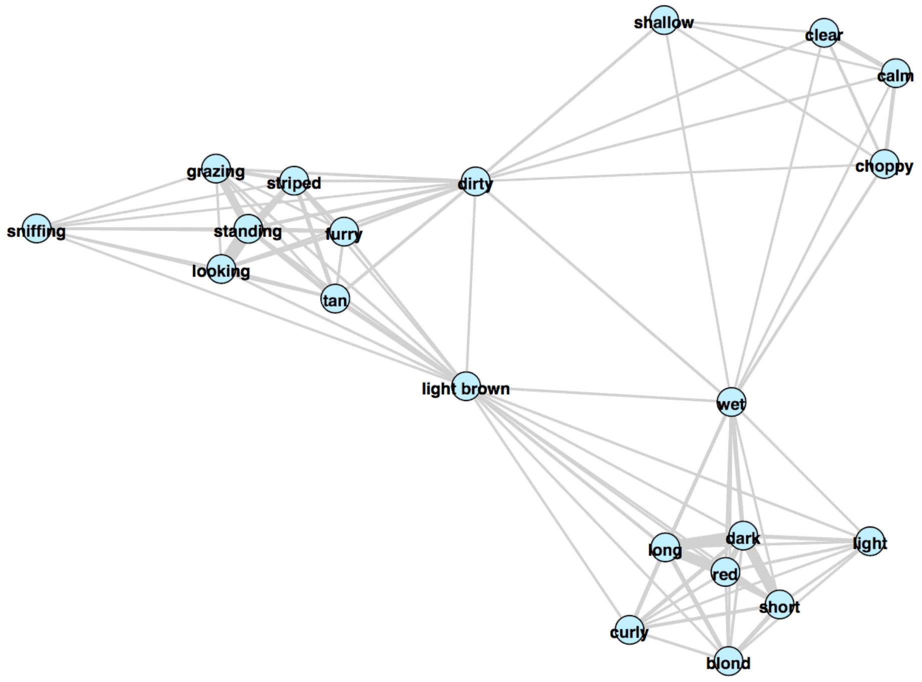


(b)

Figure 2.16: (a) Distribution showing the most common attributes in the dataset. Colors (`white`, `red`) and materials (`wooden`, `metal`) are the most common. (b) Distribution showing the number of attributes describing people. State-of-motion verbs (`standing`, `walking`) are the most common, while certain sports (`skiing`, `surfing`) are also highly represented due to an image source bias in our image set.



(a)



(b)

Figure 2.17: (a) Graph of the person-describing attributes with the most co-occurrences. Edge thickness represents the frequency of co-occurrence of the two nodes. (b) A subgraph showing the co-occurrences and intersections of three cliques, which appear to describe water (top right), hair (bottom right), and some type of animal (left). Edges between cliques have been removed for clarity.

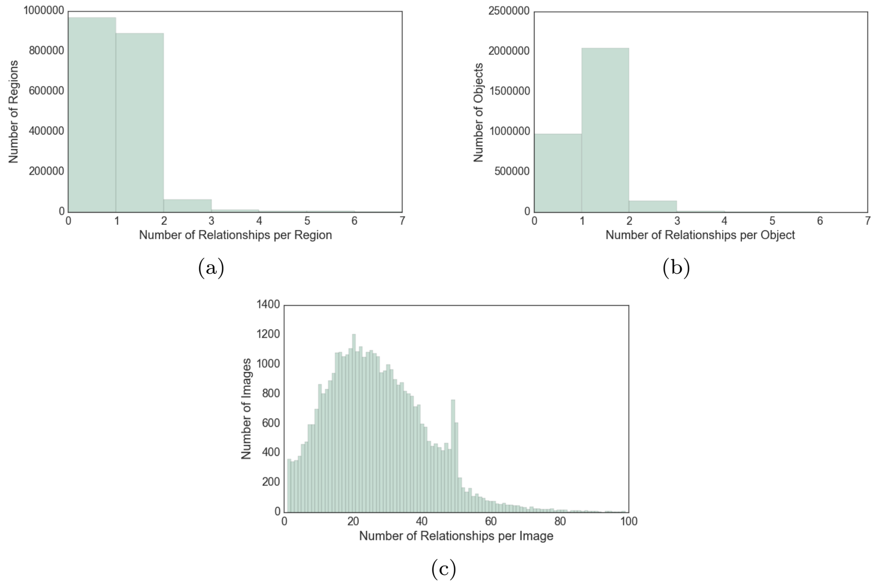


Figure 2.18: Distribution of relationships (a) per image region, (b) per image object, (c) per image.

2.2.5 Relationship Statistics

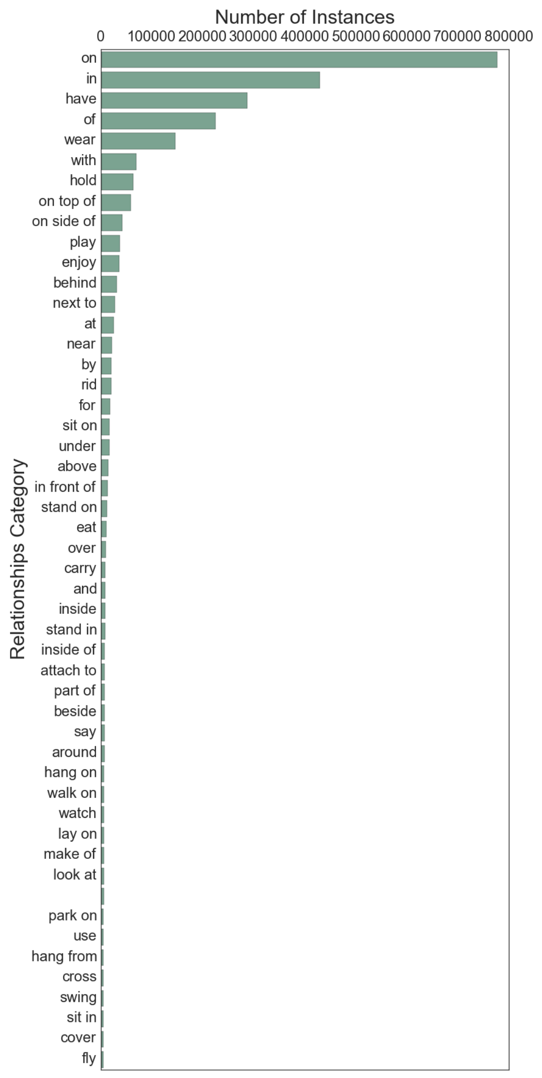
Relationships are the core components that link objects in our scene graphs. Relationships are directional, i.e. they involve two objects, one acting as the subject and one as the object of a predicate relationship. We denote all relationships in the form *relationship*(*subject*, *object*). For example, if a `man` is `swinging` a `bat`, we write *swinging*(*man*, *bat*). Relationships can be spatial (e.g. `inside_of`), action (e.g. `swinging`), compositional (e.g. `part_of`), etc. More complex relationships such as `standing_on`, which includes both an action and a spatial aspect, are also represented. Relationships are extracted from region descriptions by crowd workers, similarly to attributes and objects. Visual Genome contains a total of 13,894 unique relationships, with over 1.8 million total relationships.

Figure 2.18 (a) shows the distribution of relationships per region description. On average, we have 1 relationship per region, with a maximum of 7. We also have some descriptions like "an old, tall man," which have multiple attributes associated with the `man` but no relationships. Figure 2.18 (b) is a distribution of relationships per image object. Finally, Figure 2.18 (c) shows the distribution of relationships per image. Each image has an average of 19 relationships, with a minimum of 1 relationship and with ax maximum of over 60 relationships.

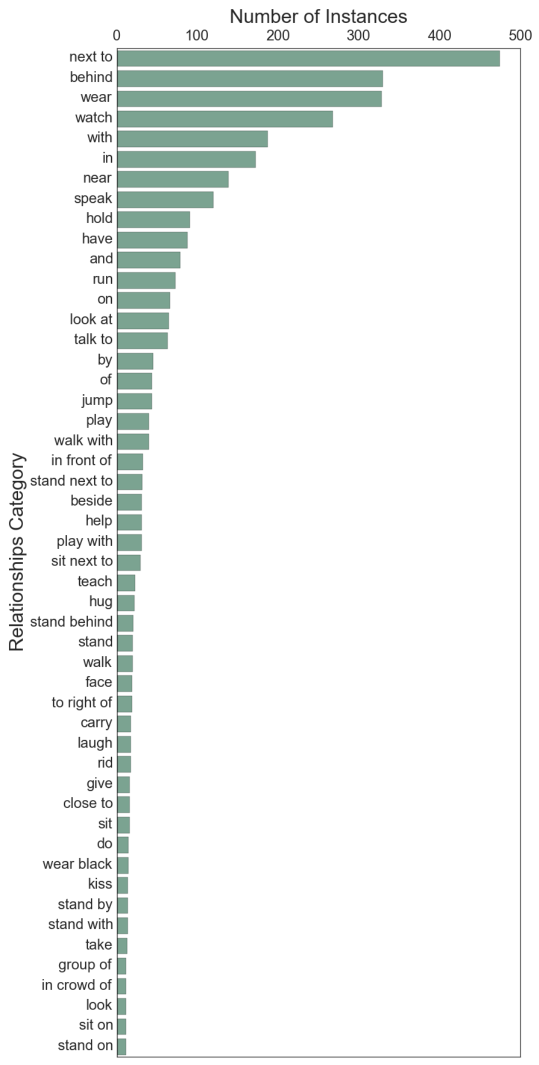
Top relationship distributions. We display the most frequently occurring relationships in Figure 2.19 (a). `on` is the most common relationship in our dataset. This is primarily because of the flexibility of the word `on`, which can refer to spatial configuration (`on top of`), attachment (`hanging on`), etc. Other common relationships involve actions like `holding` and `wearing` and spatial configurations like `behind`, `next to`, and `under`. Figure 2.19 (b) shows a similar distribution but for relationships involving people. Here we notice more human-centric relationships or actions such as `kissing`, `chatting with`, and `talking to`. The two distributions follow a Zipf distribution.

Understanding affordances. Relationships allow us to also understand the affordances of objects. We show this using a specific distribution of subjects and objects involved in the relationship `riding` in Figure 2.20. Figure 2.20 (a) shows the distribution for subjects while Figure 2.20 (b) shows a similar distribution for objects. Comparing the two distributions, we find clear patterns of people-like subject entities such as `person`, `man`, `policeman`, `boy`, and `skateboarder` that can ride other objects; the other distribution contains objects that afford `riding`, such as `horse`, `bike`, `elephant`, `motorcycle`, and `skateboard`. We can also learn specific common-sense knowledge, like that `skateboarders` only ride `skateboards` and only `surfers` ride `waves` or `surfboards`.

Related work comparison. It is also worth mentioning in this section some prior work on relationships. The concept of visual relationships has already been explored in Visual Phrases [203], who introduced a dataset of 17 such relationships such as *next_to*(*person*, *bike*) and *riding*(*person*, *horse*). However, their dataset is limited to just these 17 relationships. Similarly, the MS-COCO-a dataset [194] introduced 140 actions that humans performed in MS-COCO's dataset [139]. However, their dataset is limited to just actions, while our relationships are more general and numerous, with over 13K unique relationships. Finally, VisKE [202] introduced 6500 relationships, but in a much smaller dataset of images than Visual Genome.

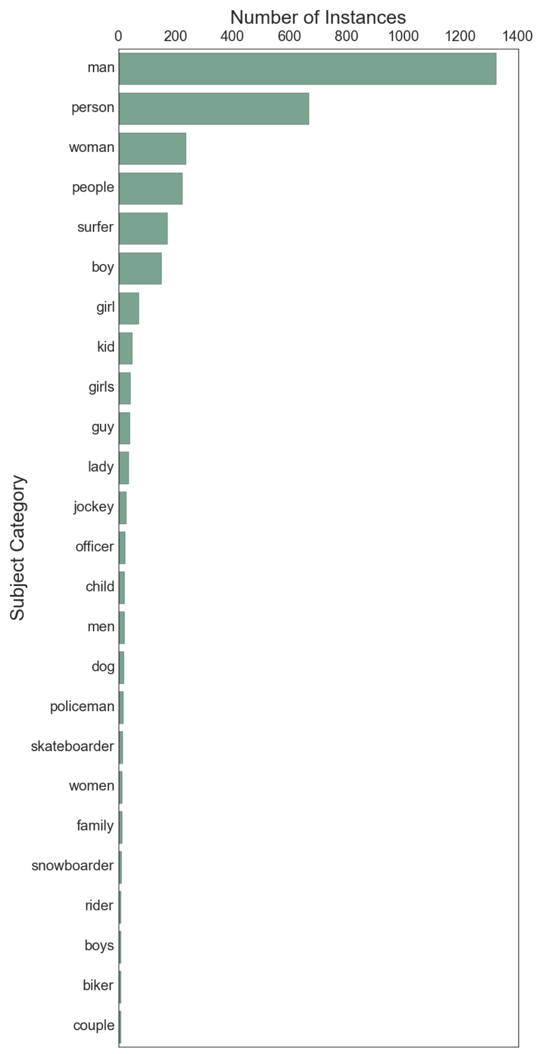


(a)

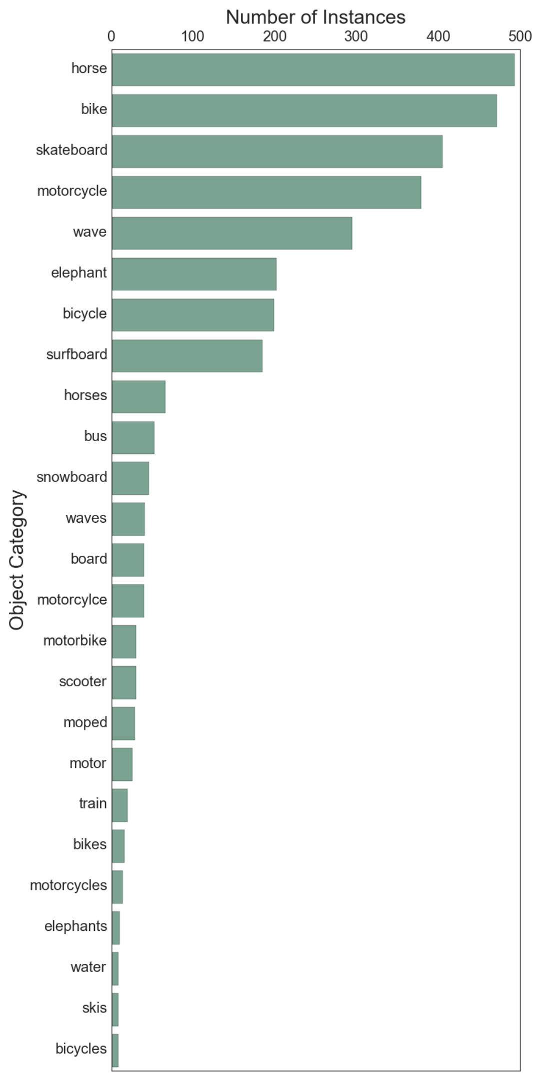


(b)

Figure 2.19: (a) A sample of the most frequent relationships in our dataset. In general, the most common relationships are spatial (`on top of`, `on side of`, etc.). (b) A sample of the most frequent relationships involving humans in our dataset. The relationships involving people tend to be more action oriented (`walk`, `speak`, `run`, etc.).



(a)



(b)

Figure 2.20: (a) Distribution of subjects for the relationship `riding`. (b) Distribution of objects for the relationship `riding`. Subjects comprise of people-like entities like `person`, `man`, `policeman`, `boy`, and `skateboarder` that can ride other objects. On the other hand, objects like `horse`, `bike`, `elephant` and `motorcycle` are entities that can afford `riding`.

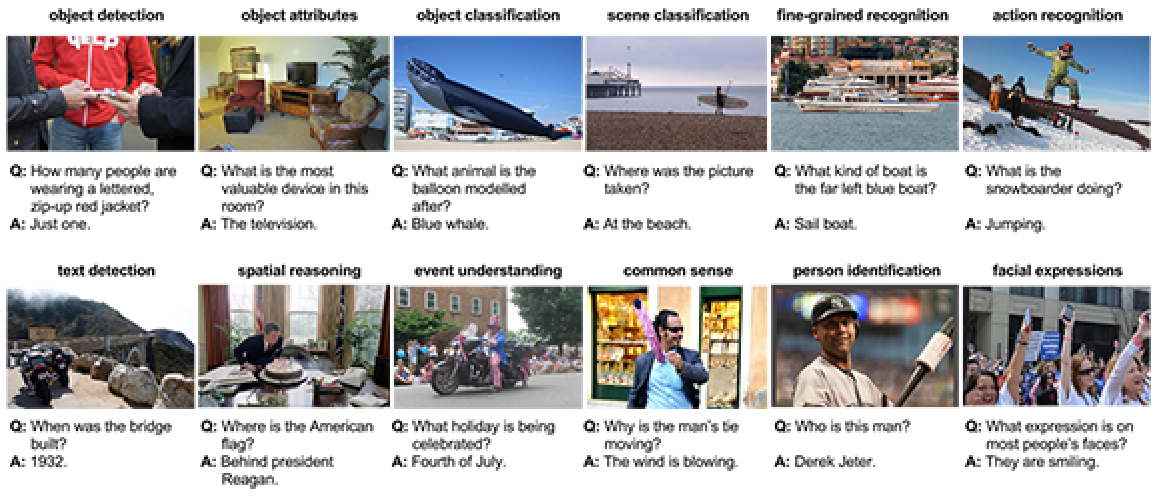

object detection
Q: How many people are wearing a lettered, zip-up red jacket?
A: Just one.
object attributes
Q: What is the most valuable device in this room?
A: The television.
object classification
Q: What animal is the balloon modelled after?
A: Blue whale.
scene classification
Q: Where was the picture taken?
A: At the beach.
fine-grained recognition
Q: What kind of boat is the far left blue boat?
A: Sail boat.
action recognition
Q: What is the snowboarder doing?
A: Jumping.
text detection
Q: When was the bridge built?
A: 1932.
spatial reasoning
Q: Where is the American flag?
A: Behind president Reagan.
event understanding
Q: What holiday is being celebrated?
A: Fourth of July.
common sense
Q: Why is the man's tie moving?
A: The wind is blowing.
person identification
Q: Who is this man?
A: Derek Jeter.
facial expressions
Q: What expression is on most people's faces?
A: They are smiling.

Figure 2.21: Example QA pairs in the Visual Genome dataset. Our QA pairs cover a spectrum of visual tasks from recognition to high-level reasoning.

	Objects	Attributes	Relationships
Region Graph	0.43	0.41	0.45
Scene Graph	21.26	16.21	18.67

Table 2.2: The average number of objects, attributes, and relationships per region graph and per scene graph.

2.2.6 Region and Scene Graph Statistics

We describe, in this thesis, the largest dataset of scene graphs to date. We use these graph representations of images as a deeper understanding of the visual world. In this section, we analyze the properties of these representations, both at the region level through region graphs and at the image level through scene graphs. We also briefly explore other datasets with scene graphs and provide aggregate statistics on our entire dataset.

Scene graphs by asking humans to write triples about an image [100]. However, unlike them, we collect graphs at a much more fine-grained level, the region graph. We obtained our graphs by asking workers to create them from the descriptions we collected from our regions. Therefore, we end up with multiple graphs for an image, one for every region description. Together, we can combine all the individual region graphs to aggregate a scene graph for an image. This scene graph is made up of all the individual region graphs. In our scene graph representation, we merge all the objects that referenced by multiple region graphs into one node in the scene graph.

Each of our images has a distribution between 40 to 50 region graphs per image, with an average of 42. Each image has exactly one scene graph. Note that the number of region descriptions and the number of region graphs for an image are not the same. For example, consider the description "it is a sunny day". Such a description contains no objects, which are the building blocks of a region graph. Therefore, such descriptions have no region graphs associated with them.

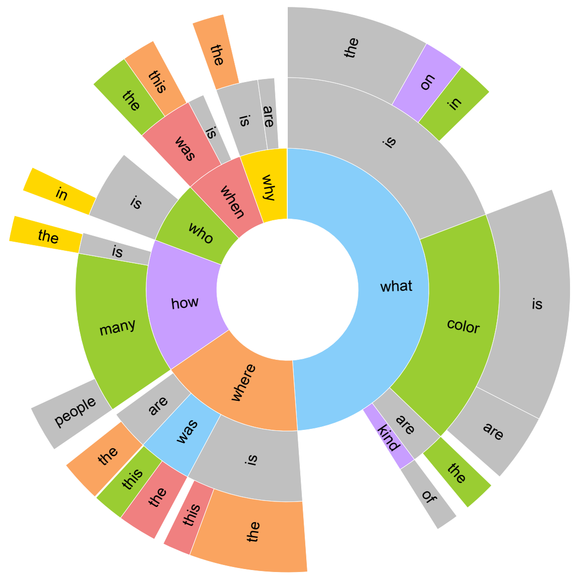
Objects, attributes, and relationships occur as a normal distribution in our data. Table 2.2 shows that in a region graph, there are an average of 0.43 objects, 0.41 attributes, and 0.45 relationships. Each scene graph and consequently each image has average of 21.26 objects, 16.21 attributes, and 18.67 relationships.

2.2.7 Question Answering Statistics

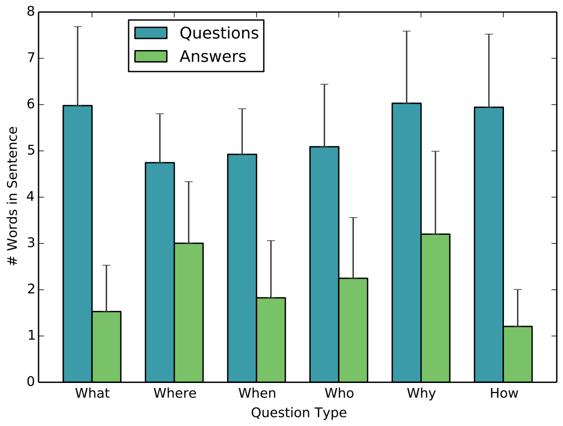
We collected $1,773,258$ question answering (QA) pairs on the Visual Genome images. Each pair consists of a question and its correct answer regarding the content of an image. On average, every image has 17 QA pairs. Rather than collecting unconstrained QA pairs as previous work has done [3, 66, 145], each question in Visual Genome starts with one of the six Ws – what, where, when, who, why, and how. There are two major benefits to focusing on six types of questions. First, they offer a considerable coverage of question types, ranging from basic perceptual tasks (e.g. recognizing objects and scenes) to complex common sense reasoning (e.g. inferring motivations of people and causality of events). Second, these categories present a natural and consistent stratification of task difficulty, indicated by the baseline performance in Chapter 5.5. For instance, *why* questions that involve complex reasoning lead to the poorest performance (3.4% top-100 accuracy) of the six categories. This enables us to obtain a better understanding of the strengths and weaknesses of today's computer vision models, which sheds light on future directions in which to proceed.

We now analyze the diversity and quality of our questions and answers. Our goal is to construct a large-scale visual question answering dataset that covers a diverse range of question types, from basic cognition tasks to complex reasoning tasks. We demonstrate the richness and diversity of our QA pairs by examining the distributions of questions and answers in Figure 2.21.

Question type distributions. The questions naturally fall into the 6W categories via their interrogative words. Inside each of the categories, the second and following words categorize the questions with increasing granularity. Inspired by VQA [3], we show the distributions of the questions by their first three words in Figure 2.22 (a). We can see that "what" is the most common of the six categories. A notable difference between our question distribution and VQA's is that we focus on ensuring that all 7 question categories are adequately represented, while in VQA, 32.37% of the questions are yes/no binary questions. As a result, a trivial model can achieve a reasonable performance by just predicting "yes" or "no" as answers. We encourage more difficult QA pairs by ruling out binary questions.



(a)



(b)

Figure 2.22: (a) Distribution of question types by starting words. This figure shows the distribution of the questions by their first three words. The angles of the regions are proportional to the number of pairs from the corresponding categories. We can see that “what” questions are the largest category with nearly half of the QA pairs. (b) Question and answer lengths by question type. The bars show the average question and answer lengths of each question type. The whiskers show the standard deviations. The factual questions, such as “what” and “how” questions, usually come with short answers of a single object or a number. This is only because “how” questions are disproportionately counting questions that start with “how many”. Questions from the “where” and “why” categories usually have phrases and sentences as answers.

Question and answer length distributions. We also analyze the question and answer lengths of each 6W category. Figure 2.22 (b) shows the average question and answer lengths of each category. Overall, the average question and answer lengths are 5.7 and 1.8 words respectively. In contrast to the VQA dataset, where .88%, 8.38%, and 3.25% of the answers consist of one, two, or three words, our answers exhibit a long-tail distribution where 57.3%, 18.1%, and 15.7% of the answers have one, two, or three words respectively. We avoid verbosity by instructing the workers to write answers as concisely as possible. The coverage of long answers means that many answers contain a short description that contains more details than merely an object or an attribute. It shows the richness and complexity of our visual QA tasks beyond object-centric recognition tasks. We foresee that these long-tail questions can motivate future research in common-sense reasoning and high-level image understanding.

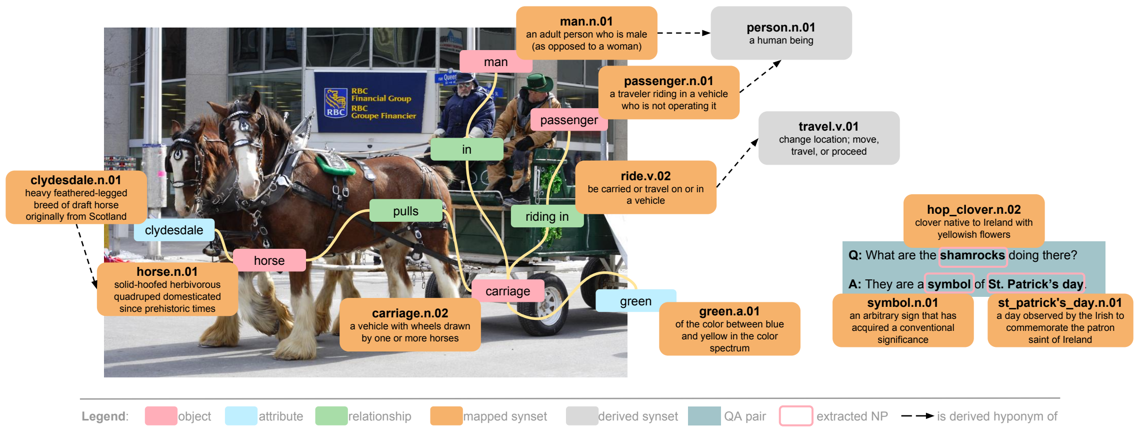

man.n.01
an adult person who is male (as opposed to a woman)
person.n.01
a human being
man
passenger.n.01
a traveler riding in a vehicle who is not operating it
passenger
travel.v.01
change location; move, travel, or proceed
in
ride.v.02
be carried or travel on or in a vehicle
clydesdale.n.01
heavy feathered-legged breed of draft horse originally from Scotland
pulls
riding in
hop_clover.n.02
clover native to Ireland with yellowish flowers
clydesdale
horse
Q: What are the shamrocks doing there?
horse.n.01
solid-hoofed herbivorous quadruped domesticated since prehistoric times
carriage
A: They are a symbol of St. Patrick's day.
green
symbol.n.01
an arbitrary sign that has acquired a conventional significance
st_patrick's_day.n.01
a day observed by the Irish to commemorate the patron saint of Ireland
carriage.n.02
a vehicle with wheels drawn by one or more horses
green.a.01
of the color between blue and yellow in the color spectrum
Legend: object attribute relationship mapped synset derived synset QA pair extracted NP is derived hyponym of

Figure 2.23: An example image from the Visual Genome dataset with its region descriptions, QA, objects, attributes, and relationships canonicalized. The large text boxes are WordNet synsets referenced by this image. For example, the `carriage` is mapped to `carriage.n.02: a vehicle with wheels drawn by one or more horses.` We do not show the bounding boxes for the objects in order to allow readers to see the image clearly. We also only show a subset of the scene graph for this image to avoid cluttering the figure.

2.2.8 Canonicalization Statistics

In order to reduce the ambiguity in the concepts of our dataset and connect it to other resources used by the research community, we canonicalize the semantic meanings of all objects, relationships, and attributes in Visual Genome. By "canonicalization," we refer to word sense disambiguation (WSD) by mapping the components in our dataset to their respective synsets in the WordNet ontology [159]. This mapping reduces the noise in the concepts contained in the dataset and also facilitates the linkage between Visual Genome and other data sources such as ImageNet [40], which is built on top of the WordNet ontology.

Figure 2.23 shows an example image from the Visual Genome dataset with its components canonicalized. For example, `horse` is canonicalized as `horse.n.01: solid-hoofed herbivorous quadruped domesticated since prehistoric times`. Its attribute, `clydesdale`, is canonicalized as its breed `clydesdale.n.01: heavy feathered-legged breed of draft horse originally from Scotland`. We also show an example of a QA from which we extract the nouns `shamrocks`, `symbol`, and `St. Patrick's day`, all of which we canonicalize to WordNet as well.

Related work. Canonicalization, or WSD [169], has been used in numerous applications, including machine translation, information retrieval, and information extraction [193, 129]. In English sentences, sentences like "He scored a goal" and "It was his goal in life" carry different meanings for

	Precision	Recall
Objects	88.0	98.5
Attributes	85.7	95.9
Relationships	92.9	88.5

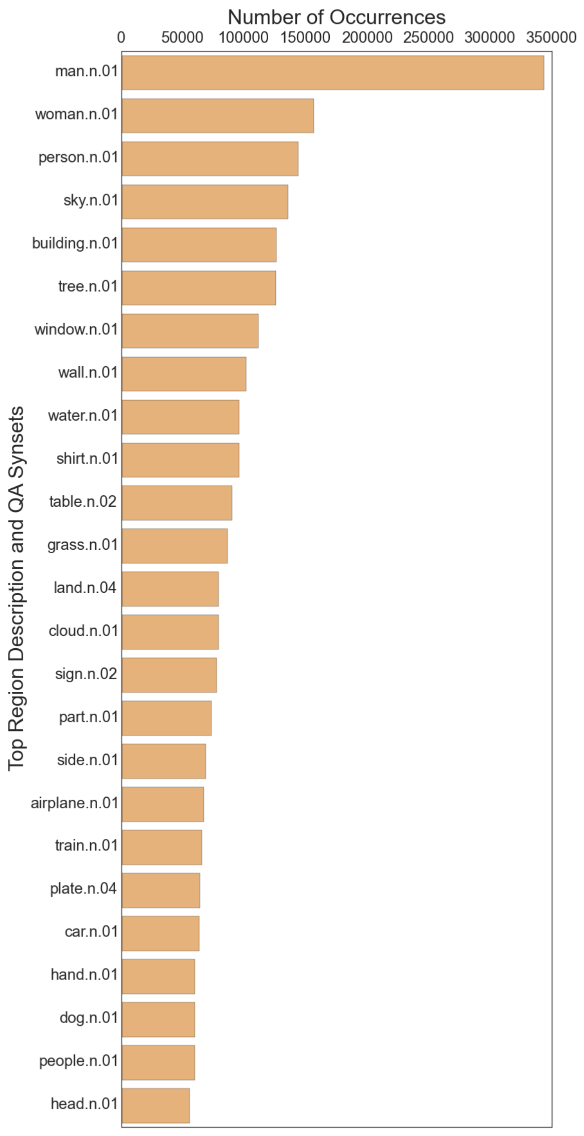
Table 2.3: Precision, recall, and mapping accuracy percentages for object, attribute, and relationship canonicalization.

the word "goal." Understanding these differences is crucial for translating languages and for returning correct results for a query. Similarly, in Visual Genome, we ensure that all our components are canonicalized to understand how different objects are related to each other; for example, "person" is a hypernym of "man" and "woman." Most past canonicalization models use precision, recall, and F1 score to evaluate on the Semeval dataset [157]. The current state-of-the-art performance on Semeval is an F1 score of 75.8% [27]. Since our canonicalization setup is different from the Semeval benchmark (we have an open vocabulary and no annotated ground truth for evaluation), our canonicalization method is not directly comparable to these existing methods. We do however, achieve a similar precision and recall score on a held-out test set described below.

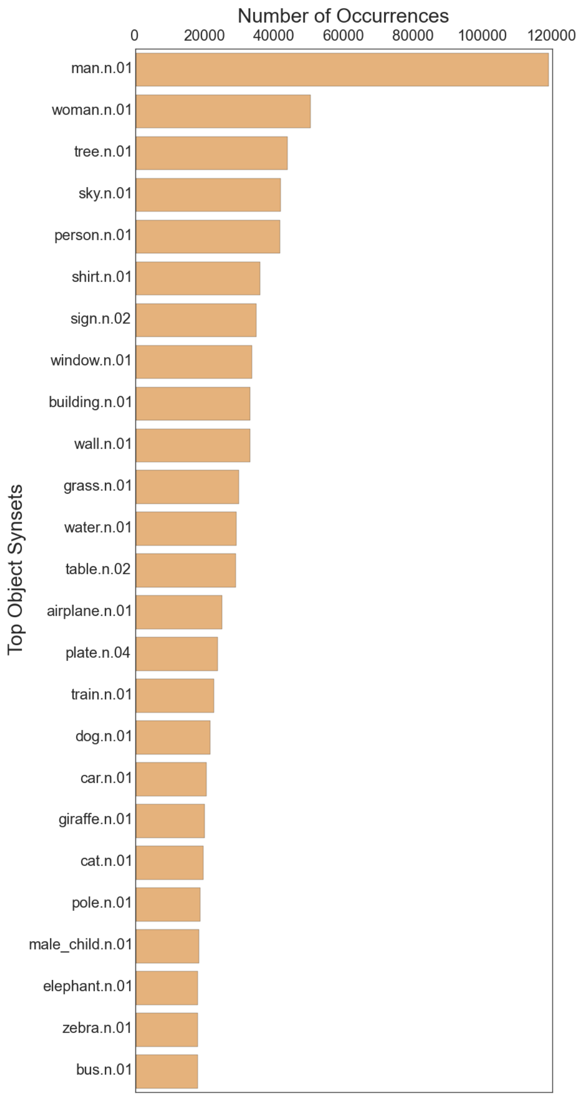
Region descriptions and QAs. We canonicalize all objects mentioned in all region descriptions and QA pairs. Because objects need to be extracted from the phrase text, we use Stanford NLP tools [148] to extract the noun phrases in each region description and QA, resulting in 99% recall of noun phrases from a subset of 200 region descriptions we manually annotated. After obtaining the noun phrases, we map each to its most frequent matching synset (according to WordNet lexeme counts). This resulted in an overall mapping accuracy of 86% and a recall of 98.5%. The most common synsets extracted from region descriptions, QAs, and objects are shown in Figure 2.24.

Attributes. We canonicalize attributes from the crowd-extracted attributes present in our scene graphs. The "attribute" designation encompasses a wide range of grammatical parts of speech. Because part-of-speech taggers rely on high-level syntax information and thus fail on the disjoint elements of our scene graphs, we normalize each attribute based on morphology alone (so-called "stemming"). Then, as with objects, we map each attribute phrase to the most frequent matching WordNet synset. We include 15 hand-mapped rules to address common failure cases in which WordNet's frequency counts prefer abstract senses of words over the spatial senses present in visual data, e.g. "short.a.01: limited in duration" over `short.a.02:  lacking in length`. For verification, we randomly sample 200 attributes, produce ground-truth mappings by hand, and compare them to the results of our algorithm. This resulted in a recall of 95.9% and a mapping accuracy of 83.5%. The most common attribute synsets are shown in Figure 2.25 (a).

Relationships. As with attributes, we canonicalize the relationships isolated in our scene graphs. We exclude prepositions, which are not recognized in WordNet, leaving a set primarily composed of verb relationships. Since the meanings of verbs are highly dependent upon their morphology and syntactic placement (e.g. passive cases, prepositional phrases), we map the structure of each relationship to the appropriate WordNet sentence frame and only consider those WordNet synsets with matching sentence frames. For each verb-synset pair, we then consider the root hypernym of that synset to reduce potential noise from WordNet's fine-grained sense distinctions. We also include 20 hand-mapped rules, again to correct for WordNet's lower representation of concrete or spatial senses; for example, the concrete `hold.v.02: have or hold in one's hand or grip` is less frequent in WordNet than the abstract `hold.v.01: cause to continue in a certain state`. For verification, we again randomly sample 200 relationships and compare the results of our canonicalization against ground-truth mappings. This resulted in a recall of 88.5% and a mapping accuracy of 77.6%. While several datasets, such as VerbNet [208] and FrameNet [4], include semantic restrictions or frames to improve classification, there is no comprehensive method of mapping to those restrictions or frames. The most common relationship synsets are shown in Figure 2.25 (b).

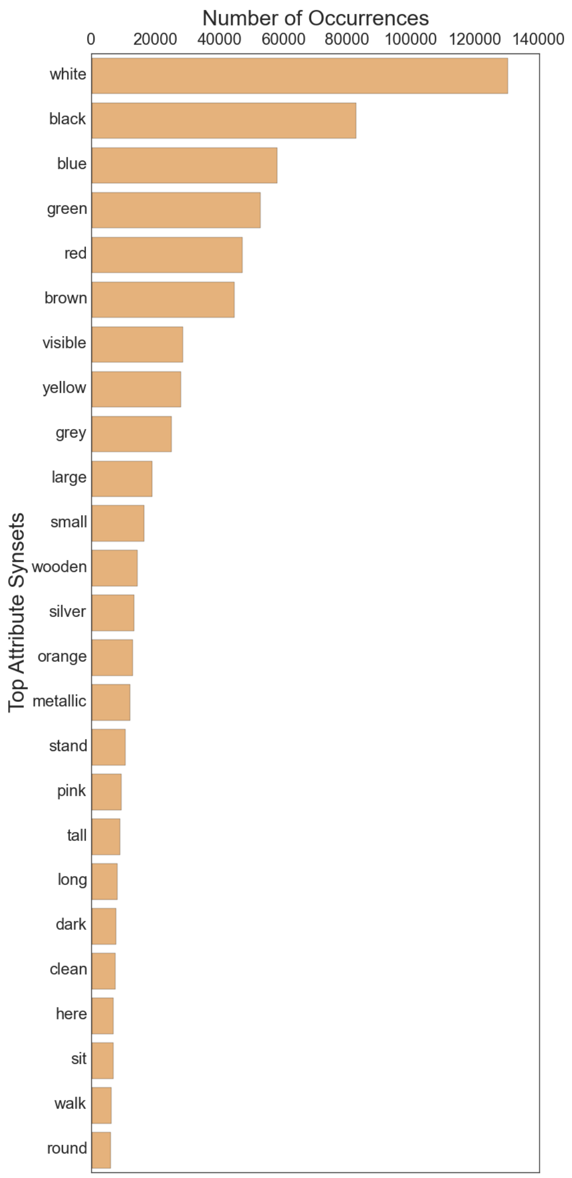


(a)

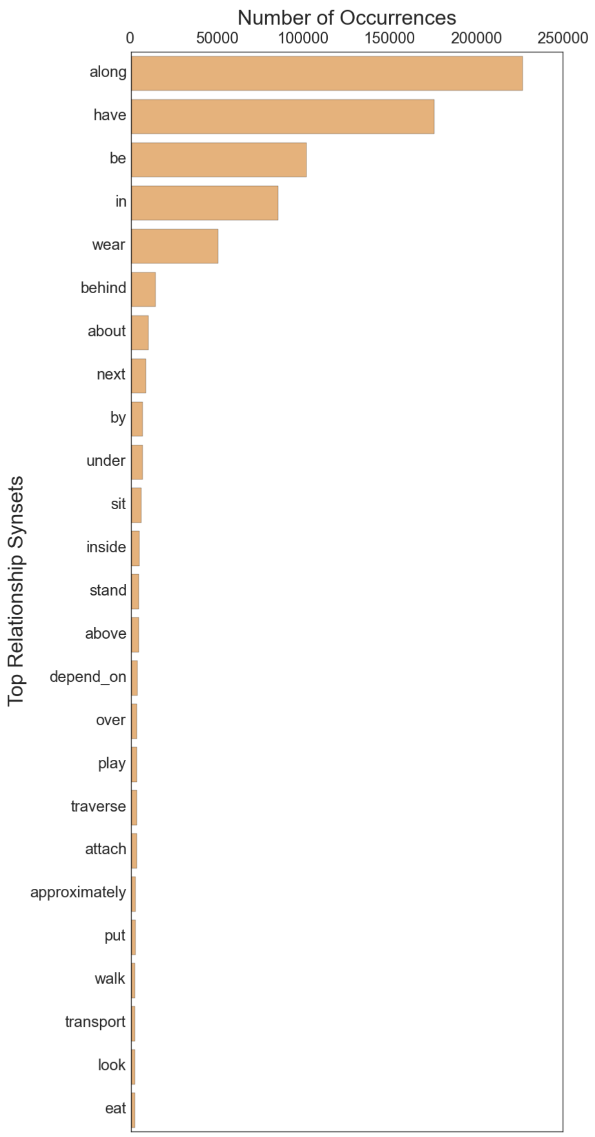


(b)

Figure 2.24: Distribution of the 25 most common synsets mapped from (a) region descriptions and question answers and (b) objects.



(a)



(b)

Figure 2.25: Distribution of the 25 most common synsets mapped from (a) attributes and (b) relationships.

Chapter 3

General Crowdsourcing Strategies

3.1 Introduction

Visual Genome was collected and verified entirely by crowd workers from Amazon Mechanical Turk. In this section, we outline the pipeline employed in creating all the components of the dataset. Each component (region descriptions, objects, attributes, relationships, region graphs, scene graphs, questions and answers) involved multiple task stages. We mention the different strategies used to make our data accurate and to enforce diversity in each component. We also provide background information about the workers who helped make Visual Genome possible.

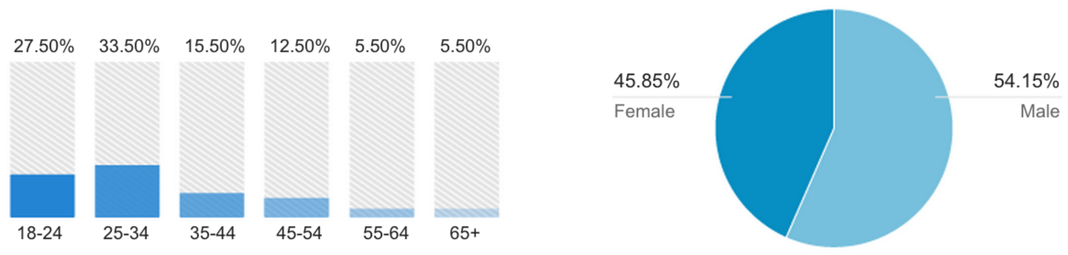
3.2 Crowd Workers

We used Amazon Mechanical Turk (AMT) as our primary source of annotations. Overall, a total of over $33,000$ unique workers contributed to the dataset. The dataset was collected over the course of 6 months after 15 months of experimentation and iteration on the data representation. Approximately $800,000$ Human Intelligence Tasks (HITs) were launched on AMT, where each HIT involved creating descriptions, questions and answers, or region graphs. Each HIT was designed such that workers manage to earn anywhere between \$6-\$8 per hour if they work continuously, in line with ethical research standards on Mechanical Turk [205]. Visual Genome HITs achieved a 94.1% retention rate, meaning that 94.1% of workers who completed one of our tasks went ahead to do more. Table 3.1 outlines the percentage distribution of the locations of the workers. 93.02% of workers contributed from the United States.

Figures 3.1 (a) and (b) outline the demographic distribution of our crowd workers. The majority of our workers were between the ages of 25 and 34 years old. Our youngest contributor was 18 years old and the oldest was 68 years old. We also had a near-balanced split of 54.15% male and 45.85% female workers.

Country	**Distribution**
United States	93.02%
Philippines	1.29%
Kenya	1.13%
India	0.94%
Russia	0.50%
Canada	0.47%
(Others)	2.65%

Table 3.1: Geographic distribution of countries from where crowd workers contributed to Visual Genome.



(a) (b)

Figure 3.1: (a) Age and (b) gender distribution of Visual Genome's crowd workers.

3.3 Region Descriptions

Visual Genome's main goal is to enable the study of cognitive computer vision tasks. The next step towards understanding images requires studying relationships between objects in scene graph representations of images. However, we observed that collecting scene graphs directly from an image leads to workers annotating easy, frequently-occurring relationships like *wearing*(*man*, *shirt*) instead of focusing on salient parts of the image. This is evident from previous datasets [100, 142] that contain a large number of such relationships. After experimentation, we observed that when asked to describe an image using natural language, crowd workers naturally start with the most salient part of the image and then move to describing other parts of the image one by one. Inspired by this finding, we focused our attention towards collecting a dataset of region descriptions that is diverse in content.

When a new image is added to the crowdsourcing pipeline with no annotations, it is sent to a worker who is asked to draw three bounding boxes and write three descriptions for the region enclosed by each box. Next, the image is sent to another worker along with the previously written descriptions. Workers are explicitly encouraged to write descriptions that have not been written before. This process is repeated until we have collect 50 region descriptions for each image. To prevent workers from having to skim through a long list of previously written descriptions, we only show them the top seven most similar descriptions. We calculate these most similar descriptions using BLEU [171] (n-gram) scores between pairs of sentences. We define the BLEU score between a description d_i and a previous description d_j to be:

$$BLEU_N(d_i, d_j) = b(d_i, d_j)\exp(\frac{1}{N}\sum_{n=1}^{N}\log p_n(d_i, d_j)) \tag{3.1}$$

where we enforce a brevity penalty using:

$$b(d_i, d_j) = \begin{cases} 1 & \text{if } len(d_i) > len(d_j) \\ e^{1-\frac{len(d_j)}{len(d_i)}} & \text{otherwise} \end{cases} \tag{3.2}$$

and p_n calculates the percentage of n-grams in d_i that match n-grams in d_j.

When a worker writes a new description, we programmatically enforce that it has not been repeated by using BLEU score thresholds set to 0.7 to ensure that it is dissimilar to descriptions from both of the following two lists:

1. **Image-specific descriptions.** A list of all previously written descriptions for that image.
2. **Global image descriptions.** A list of the top 100 most common written descriptions of all images in the dataset. This prevents very common phrases like "sky is blue" from dominating the set of region descriptions.

Finally, we ask workers to draw bounding boxes that satisfy one requirement: **coverage**. The bounding box must cover all objects mentioned in the description. Figure 3.2 shows an example of a good box that covers both the `street` as well the `car` mentioned in the description, as well as an example of a bad box.

3.4 Objects

Once 50 region descriptions are collected for an image, we extract the visual objects from each description. Each description is sent to one crowd worker, who extracts all the objects from the description and grounds each object as a bounding box in the image. For example, from Figure 2.1, let's consider the description "woman in shorts is standing behind the man." A worker would extract three objects: `woman`, `shorts`, and `man`. They would then draw a box around each of the objects. We require each bounding box to be drawn to satisfy two requirements: **coverage** and **quality**. Coverage has the same definition as described above in Section 3.3, where we ask workers to make sure that the bounding box covers the object completely (Figure 3.3). Quality requires that each bounding box be as tight as possible around its object such that if the box's length or height were

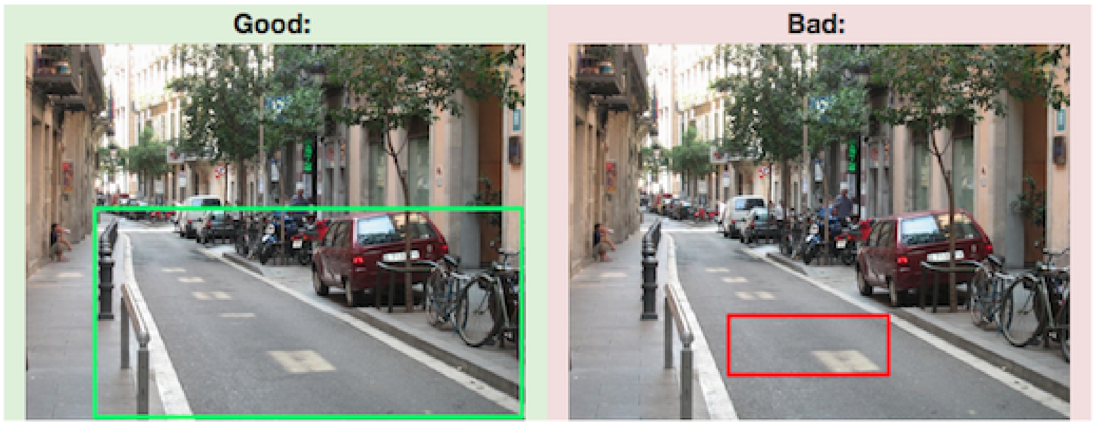


Figure 3.2: Good (left) and bad (right) bounding boxes for the phrase "a street with a red car parked on the side," judged on **coverage**.

decreased by one pixel, it would no longer satisfy the coverage requirement. Since a one pixel error can be physically impossible for most workers, we relax the definition of quality to four pixels.

Multiple descriptions for an image might refer to the same object, sometimes with different words. For example, a `man` in one description might be referred to as `person` in another description. We can thus use this crowdsourcing stage to build these co-reference chains. With each region description given to a worker to process, we include a list of previously extracted objects as suggestions. This allows a worker to choose a previously drawn box annotated as `man` instead of redrawing a new box for `person`.

Finally, to increase the speed with which workers complete this task, we also use Stanford's dependency parser [148] to extract nouns automatically and send them to the workers as suggestions. While the parser manages to find most of the nouns, it sometimes misses compound nouns, so we avoided completely depending on this automated method. By combining the parser with crowdsourcing tasks, we were able to speed up our object extraction process without losing accuracy.

3.5 Attributes, Relationships, and Region Graphs

Once all objects have been extracted from each region description, we can extract the attributes and relationships described in the region. We present each worker with a region description along with its extracted objects and ask them to add attributes to objects or to connect pairs of objects with relationships, based on the text of the description. From the description "woman in shorts is standing behind the man", workers will extract the attribute `standing` for the `woman` and the relationships *in*(*woman*, *shorts*) and *behind*(*woman*, *man*). Together, objects, attributes, and relationships form the region graph for a region description. Some descriptions like "it is a sunny day" do not contain any objects and therefore have no region graphs associated with them. Workers

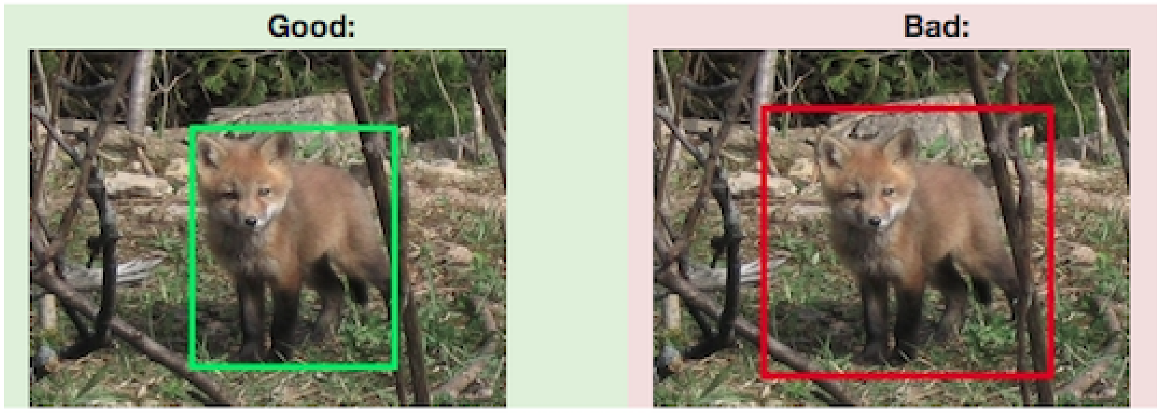


Figure 3.3: Good (left) and bad (right) bounding boxes for the object `fox`, judged on both **coverage** as well as **quality**.

.

are asked to not generate any graphs for such descriptions. We create scene graphs by combining all the region graphs for an image by combining all the co-referenced objects from different region graphs.

3.6 Scene Graphs

The scene graph is the union of all region graphs extracted from region descriptions. We merge nodes from region graphs that correspond to the same object; for example, `man` and `person` in two different region graphs might refer to the same object in the image. We say that objects from different graphs refer to the same object if their bounding boxes have an overlap over union of 0.8. However, this heuristic might contain false positives. So, before merging two objects, we ask workers to confirm that a pair of objects with significant overlap are indeed the same object. For example, in Figure 3.4 (right), the `fox` might be extracted from two different region descriptions. These boxes are then combined together (Figure 3.4 (left)) when constructing the scene graph. Two region graphs are combined together by merging objects that are co-referenced by both the graphs.

3.7 Questions and Answers

To create question answer (QA) pairs, we ask the AMT workers to write pairs of questions and answers about an image. To ensure quality, we instruct the workers to follow three rules: 1) start the questions with one of the "seven Ws" (`who`, `what`, `where`, `when`, `why`, `how` and `which`); 2) avoid ambiguous and speculative questions; 3) be precise and unique, and relate the question to the image such that it is clearly answerable if and only if the image is shown.

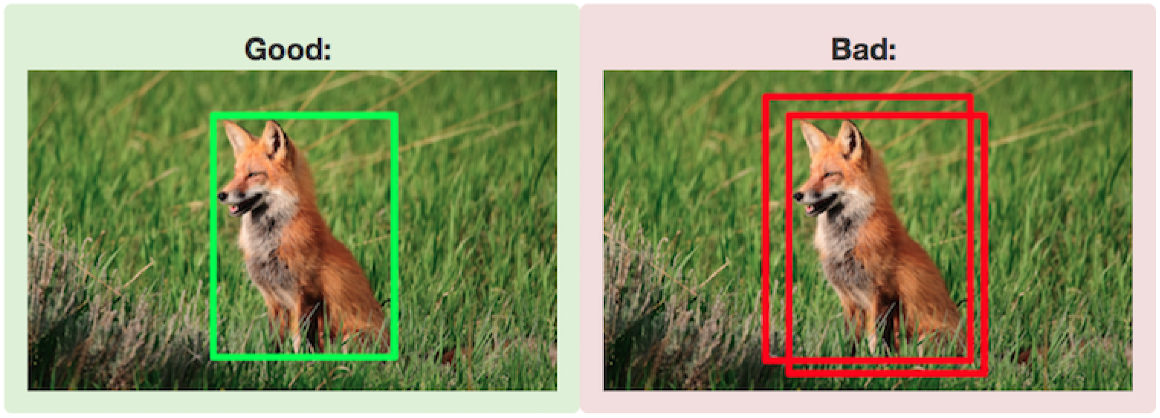


Figure 3.4: Each object (`fox`) has only one bounding box referring to it (left). Multiple boxes drawn for the same object (right) are combined together if they have a minimum threshold of 0.9 intersection over union.

.

We collected two separate types of QAs: freeform QAs and region-based QAs. In freeform QA, we ask a worker to look at an image and write eight QA pairs about it. To encourage diversity, we enforce that workers write at least three different Ws out of the seven in their eight pairs. In region-based QA, we ask the workers to write a pair based on a given region. We select the regions that have large areas (more than 5k pixels) and long phrases (more than 4 words). This enables us to collect around twenty region-based pairs at the same cost of the eight freeform QAs. In general, freeform QA tends to yield more diverse QA pairs that enrich the question distribution; region-based QA tends to produce more factual QA pairs at a lower cost.

3.8 Verification

All Visual Genome data go through a verification stage as soon as they are annotated. This stage helps eliminate incorrectly labeled objects, attributes, and relationships. It also helps remove region descriptions and questions and answers that might be correct but are vague ("This person seems to enjoy the sun."), subjective ("room looks dirty"), or opinionated ("Being exposed to hot sun like this may cause cancer").

Verification is conducted using two separate strategies: majority voting [221] and rapid judgments [113]. All components of the dataset except objects are verified using majority voting. Majority voting [221] involves three unique workers looking at each annotation and voting on whether it is factually correct. An annotation is added to our dataset if at least two (a majority) out of the three workers verify that it is correct.

We only use rapid judgments to speed up the verification of the objects in our dataset. Meanwhile, rapid judgments [113] use an interface inspired by rapid serial visual processing that enable verification of objects with an order of magnitude increase in speed than majority voting.

3.9 Canonicalization

All the descriptions and QAs that we collect are freeform worker-generated texts. They are not constrained by any limitations. For example, we do not force workers to refer to a man in the image as a `man`. We allow them to choose to refer to the man as `person`, `boy`, `man`, etc. This ambiguity makes it difficult to collect all instances of `man` from our dataset. In order to reduce the ambiguity in the concepts of our dataset and connect it to other resources used by the research community, we map all objects, attributes, relationships, and noun phrases in region descriptions and QAs to synsets in WordNet [159]. In the example above, `person`, `boy`, and `man` would map to the synsets: `person.n.01 (a human being)`, `male_child.n.01 (a youthful male person)` and `man.n.03 (the generic use of the word to refer to any human being)` respectively. Thanks to the WordNet hierarchy it is now possible to fuse those three expressions of the same concept into `person.n.01 (a human being)` since this is the lowest common ancestor node of all aforementioned synsets.

We use the Stanford NLP tools [148] to extract the noun phrases from the region descriptions and QAs. Next, we map them to their most frequent matching synset in WordNet according to WordNet lexeme counts. We then refine this simple heuristic by hand-crafting mapping rules for the 30 most common failure cases. For example according to WordNet's lexeme counts the most common semantic for "table" is `table.n.01 (a set of data arranged in rows and columns)`. However in our data it is more likely to see pieces of furniture and therefore bias the mapping towards `table.n.02 (a piece of furniture having a smooth flat top that is usually supported by one or more vertical legs)`. The objects in our scene graphs are already noun phrases and are mapped to WordNet in the same way.

We normalize each attribute based on morphology (so called "stemming") and map them to the WordNet adjectives. We include 15 hand-crafted rules to address common failure cases, which typically occur when the concrete or spatial sense of the word seen in an image is not the most common overall sense. For example, the synset `long.a.02 (of relatively great or greater than average spatial extension)` is less common in WordNet than `long.a.01 (indicating a relatively great or greater than average duration of time)`, even though instances of the word "long" in our images are much more likely to refer to that spatial sense.

For relationships, we ignore all prepositions as they are not recognized by WordNet. Since the meanings of verbs are highly dependent upon their morphology and syntactic placement (e.g. passive cases, prepositional phrases), we try to find WordNet synsets whose sentence frames match with the context of the relationship. Sentence frames in WordNet are formalized syntactic frames in which a certain sense of a word might appear; for example, `play.v.01: participate in games or sport` occurs in the sentence frames "Somebody [play]s" and "Somebody [play]s something." For each verb-synset pair, we then consider the root hypernym of that synset to reduce potential noise from WordNet's fine-grained sense distinctions. The WordNet hierarchy for verbs is segmented and originates from over 100 root verbs. For example, `draw.v.01: cause to move by pulling` traces back to the root hypernym `move.v.02: cause to move or shift into a new position`, while `draw.v.02: get or derive` traces to the root `get.v.01: come into the possession of something concrete or abstract`. We also include 20 hand-mapped rules, again to correct for WordNet's lower representation of concrete or spatial senses.

These mappings are not perfect and still contain some ambiguity. Therefore, we send all our mappings along with the top four alternative synsets for each term to Amazon Mechanical Turk. We ask workers to verify that our mapping was accurate and change the mapping to an alternative one if it was a better fit. We present workers with the concept we want to canonicalize along with our proposed corresponding synset with 4 additional options. To prevent workers from always defaulting to the our proposed synset, we do not explicitly specify which one of the 5 synsets presented is our proposed synset. Section 2.2.8 provides experimental precision and recall scores for our canonicalization strategy.

Chapter 4

Embracing Error to Enable Rapid Crowdsourcing

4.1 Introduction

Social science [110, 154], interactive systems [51, 119] and machine learning [40, 139] are becoming more and more reliant on large-scale, human-annotated data. Increasingly large annotated datasets have unlocked a string of social scientific insights [69, 19] and machine learning performance improvements [116, 71, 242]. One of the main enablers of this growth has been microtask crowdsourcing [221]. Microtask crowdsourcing marketplaces such as Amazon Mechanical Turk offer a scale and cost that makes such annotation feasible. As a result, companies are now using crowd work to complete hundreds of thousands of tasks per day [151].

However, even microtask crowdsourcing can be insufficiently scalable, and it remains too expensive for use in the production of many industry-size datasets [101]. Cost is bound to the amount of work completed per minute of effort, and existing techniques for speeding up labeling (reducing the amount of required effort) are not scaling as quickly as the volume of data we are now producing that must be labeled [236]. To expand the applicability of crowdsourcing, the number of items annotated per minute of effort needs to increase substantially.

In this chapter, we focus on one of the most common classes of crowdsourcing tasks [90]: binary annotation. These tasks are yes-or-no questions, typically identifying whether or not an input has a specific characteristic. Examples of these types of tasks are topic categorization (e.g., "Is this article about finance?") [207], image classification (e.g., "Is this a dog?") [40, 139, 136], audio styles [210] and emotion detection [136] in songs (e.g., "Is the music calm and soothing?"), word similarity (e.g., "Are *shipment* and *cargo* synonyms?") [160] and sentiment analysis (e.g., "Is this tweet positive?") [170].

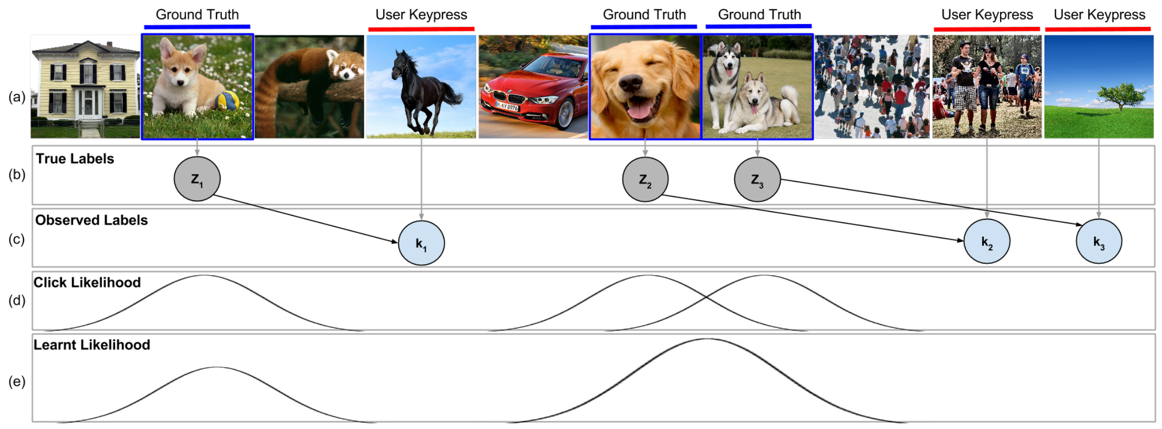


Figure 4.1: (a) Images are shown to workers at 100ms per image. Workers react whenever they see a dog. (b) The true labels are the ground truth dog images. (c) The workers' keypresses are slow and occur several images after the dog images have already passed. We record these keypresses as the observed labels. (d) Our technique models each keypress as a delayed Gaussian to predict (e) the probability of an image containing a dog from these observed labels.

Previous methods have sped up binary classification tasks by minimizing worker error.A central assumption behind this prior work has been that workers make errors because they are not trying hard enough (e.g., "a lack of expertise, dedication [or] interest" [212]). Platforms thus punish errors harshly, for example by denying payment. Current methods calculate the minimum redundancy necessary to be confident that errors have been removed [212, 219, 220]. These methods typically result in a 0.25× to 1× speedup beyond a fixed majority vote [175, 199, 212, 105].

We take the opposite position: that designing the task to encourage some error, or even make errors inevitable, can produce far greater speedups.Because platforms strongly punish errors, workers carefully examine even straightforward tasks to make sure they do not represent edge cases [153, 92]. The result is slow, deliberate work. We suggest that there are cases where we can encourage workers to move quickly by telling them that making some errors is acceptable.Though individual worker accuracy decreases, we can recover from these mistakes post-hoc algorithmically (Figure 4.1).

We manifest this idea via a crowdsourcing technique in which workers label a rapidly advancing stream of inputs. Workers are given a binary question to answer, and they observe as the stream automatically advances via a method inspired by rapid serial visual presentation (RSVP) [134, 54]. Workers press a key whenever the answer is "yes" for one of the stream items. Because the stream is advancing rapidly, workers miss some items and have delayed responses. However, workers are reassured that the requester expects them to miss a few items. To recover the correct answers, the technique randomizes the item order for each worker and model workers' delays as a normal distribution whose variance depends on the stream's speed. For example, when labeling whether images have a "barking dog" in them, a self-paced worker on this task takes 1.7s per image on average. With our technique, workers are shown a stream at 100ms per image. The technique models the delays experienced at different input speeds and estimates the probability of intended labels from the key presses.

We evaluate our technique by comparing the total worker time necessary to achieve the same precision on an image labeling task as a standard setup with majority vote. The standard approach takes three workers an average of 1.7s each for a total of 5.1s. Our technique achieves identical precision (97%) with five workers at 100ms each, for a total of 500ms of work. The result is an order of magnitude speedup of 10×.

This relative improvement is robust across both simple tasks, such as identifying dogs, and complicated tasks, such as identifying "a person riding a motorcycle" (interactions between two objects) or "people eating breakfast" (understanding relationships among many objects). We generalize our technique to other tasks such as word similarity detection, topic classification and sentiment analysis. Additionally, we extend our method to categorical classification tasks through a ranked cascade of binary classifications. Finally, we test workers' subjective mental workload and find no measurable increase.

Contributions. We make the following contributions:

1. We introduce a rapid crowdsourcing technique that makes errors normal and even inevitable. We show that it can be used to effectively label large datasets by achieving a speedup of an order of magnitude on several binary labeling crowdsourcing tasks.
2. We demonstrate that our technique can be generalized to multi-label categorical labeling tasks, combined independently with existing optimization techniques, and deployed without increasing worker mental workload.

4.2 Related Work

The main motivation behind our work is to provide an environment where humans can make decisions quickly. We encourage a margin of human error in the interface that is then rectified by inferring the true labels algorithmically. In this section, we review prior work on crowdsourcing optimization and other methods for motivating contributions. Much of this work relies on artificial intelligence techniques: we complement this literature by changing the crowdsourcing interface rather than focusing on the underlying statistical model.

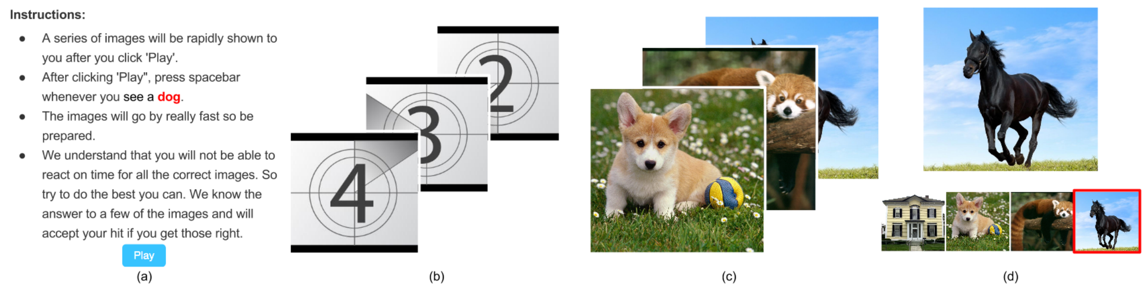
Our technique is inspired by rapid serial visual presentation (RSVP), a technique for consuming media rapidly by aligning it within the foveal region and advancing between items quickly [134, 54]. RSVP has already been proven to be effective at speeding up reading rates [252]. RSVP users can react to a single target image in a sequence of images even at 125ms per image with 75% accuracy [181]. However, when trying to recognize concepts in images, RSVP only achieves an accuracy of 10% at the same speed [182]. In our work, we integrate multiple workers' errors to successfully extract true labels.

Many previous papers have explored ways of modeling workers to remove bias or errors from ground truth labels [251, 249, 267, 175, 91]. For example, an unsupervised method for judging worker quality can be used as a prior to remove bias on binary verification labels [91]. Individual workers can also be modeled as projections into an open space representing their skills in labeling a particular image [251]. Workers may have unknown expertise that may in some cases prove adversarial to the task. Such adversarial workers can be detected by jointly learning the difficulty of labeling a particular datum along with the expertises of workers [249]. Finally, a generative model can be used to model workers' skills by minimizing the entropy of the distribution over their labels and the unknown true labels [267]. We draw inspiration from this literature, calibrating our model using a similar generative approach to understand worker reaction times. We model each worker's reaction as a delayed Gaussian distribution.

In an effort to reduce cost, many previous papers have studied the tradeoffs between speed (cost) and accuracy on a wide range of tasks [246, 15, 244, 198]. Some methods estimate human time with annotation accuracy to jointly model the errors in the annotation process [246, 15, 244]. Other methods vary both the labeling cost and annotation accuracy to calculate a tradeoff between the two [95, 41]. Similarly, some crowdsourcing systems optimize a budget to measure confidence in worker annotations [104, 105]. Models can also predict the redundancy of non-expert labels needed to match expert-level annotations [212]. Just like these methods, we show that non-experts can use our technique and provide expert-quality annotations; we also compare our methods to the conventional majority-voting annotation scheme.

Another perspective on rapid crowdsourcing is to return results in real time, often by using a retainer model to recall workers quickly [9, 128, 126]. Like our approach, real-time crowdsourcing can use algorithmic solutions to combine multiple in-progress contributions [127]. These systems' techniques could be fused with ours to create crowds that can react to bursty requests.

One common method for optimizing crowdsourcing is active learning, which involves learning algorithms that interactively query the user. Examples include training image recognition [226] and attribution recognition [173] with fewer examples. Comparative models for ranking attribute models have also optimized crowdsourcing using active learning [137]. Similar techniques have explored optimization of the "crowd kernel" by adaptively choosing the next questions asked of the crowd in order to build a similarity matrix between a given set of data points [232]. Active learning needs to decide on a new task after each new piece of data is gathered from the crowd. Such models tend to be quite expensive to compute. Other methods have been proposed to decide on a set of tasks instead of just one task [241]. We draw on this literature: in our technique, after all the images have been seen by at least one worker, we use active learning to decide the next set of tasks. We determine which images to discard and which images to group together and send this set to another



Instructions:

- A series of images will be rapidly shown to you after you click 'Play'.
- After clicking 'Play", press spacebar whenever you see a **dog**.
- The images will go by really fast so be prepared.
- We understand that you will not be able to react on time for all the correct images. So try to do the best you can. We know the answer to a few of the images and will accept your hit if you get those right.

Figure 4.2: (a) Task instructions inform workers that we expect them to make mistakes since the items will be displayed rapidly. (b) A string of countdown images prepares them for the rate at which items will be displayed. (c) An example image of a "dog" shown in the stream—the two images appearing behind it are included for clarity but are not displayed to workers. (d) When the worker presses a key, we show the last four images below the stream of images to indicate which images might have just been labeled.

worker to gather more information.

Finally, there is a group of techniques that attempt to optimize label collection by reducing the number of questions that must be answered by the crowd. For example, a hierarchy in label distribution can reduce the annotation search space [41], and information gain can reduce the number of labels necessary to build large taxonomies using a crowd [29, 13]. Methods have also been proposed to maximize accuracy of object localization in images [229] and videos [243]. Previous labels can also be used as a prior to optimize acquisition of new types of annotations [14]. One of the benefits of our technique is that it can be used independently of these others to jointly improve crowdsourcing schemes. We demonstrate the gains of such a combination in our evaluation.

4.3 Error-Embracing Crowdsourcing

Current microtask crowdsourcing platforms like Amazon Mechanical Turk incentivize workers to avoid rejections [92, 153], resulting in slow and meticulous work. But is such careful work necessary to build an accurate dataset? In this section, we detail our technique for rapid crowdsourcing by encouraging less accurate work.

The design space of such techniques must consider which tradeoffs are acceptable to make. The first relevant dimension is accuracy. When labeling a large dataset (e.g., building a dataset of ten thousand articles about housing), *precision* is often the highest priority: articles labeled as on-topic by the system must in fact be about housing. *Recall*, on the other hand, is often less important, because there is typically a large amount of available unlabeled data: even if the system misses some on-topic articles, the system can label more items until it reaches the desired dataset size. We thus develop an approach for producing high precision at high speed, sacrificing some recall if necessary.

The second design dimension involves the task characteristics. Many large-scale crowdsourcing tasks involve closed-ended responses such as binary or categorical classifications. These tasks have two useful properties. First, they are time-bound by users' perception and cognition speed rather than motor (e.g., pointing, typing) speed [28], since acting requires only a single button press. Second, it is possible to aggregate responses automatically, for example with majority vote. Open-ended crowdsourcing tasks such as writing [10] or transcription are often time-bound by data entry motor speeds and cannot be automatically aggregated. Thus, with our technique, we focus on closed-ended tasks.

4.3.1 Rapid crowdsourcing of binary decision tasks

Binary questions are one of the most common classes of crowdsourcing tasks. Each yes-or-no question gathers a label on whether each item has a certain characteristic. In our technique, rather than letting workers focus on each item too carefully, we display each item for a specific period of time before moving on to the next one in a rapid slideshow. For example, in the context of an image verification task, we show workers a stream of images and ask them to press the spacebar whenever they see a specific class of image. In the example in Figure 4.2, we ask them to react whenever they see a "dog."

The main parameter in this approach is the length of time each item is visible. To determine the best option, we begin by allowing workers to work at their own pace. This establishes an initial average time period, which we then slowly decrease in successive versions until workers start making mistakes [28]. Once we have identified this error point, we can algorithmically model workers' latency and errors to extract the true labels.

To avoid stressing out workers, it is important that the task instructions convey the nature of the rapid task and the fact that we expect them to make some errors. Workers are first shown a set of instructions (Figure 4.2(a)) for the task. They are warned that reacting to every single correct image on time is not feasible and thus not expected. We also warn them that we have placed a small number of items in the set that we know to be positive items. These help us calibrate each worker's speed and also provide us with a mechanism to reject workers who do not react to any of the items.

Once workers start the stream (Figure 4.2(b)), it is important to prepare them for pace of the task. We thus show a film-style countdown for the first few seconds that decrements to zero at the same interval as the main task. Without these countdown images, workers use up the first few seconds getting used to the pace and speed. Figure 4.2(c) shows an example "dog" image that is displayed in front of the user. The dimensions of all items (images) shown are held constant to avoid having to adjust to larger or smaller visual ranges.

When items are displayed for less than 400ms, workers tend to react to all positive items with a delay. If the interface only reacts with a simple confirmation when workers press the spacebar, many workers worry that they are too late because another item is already on the screen. Our solution

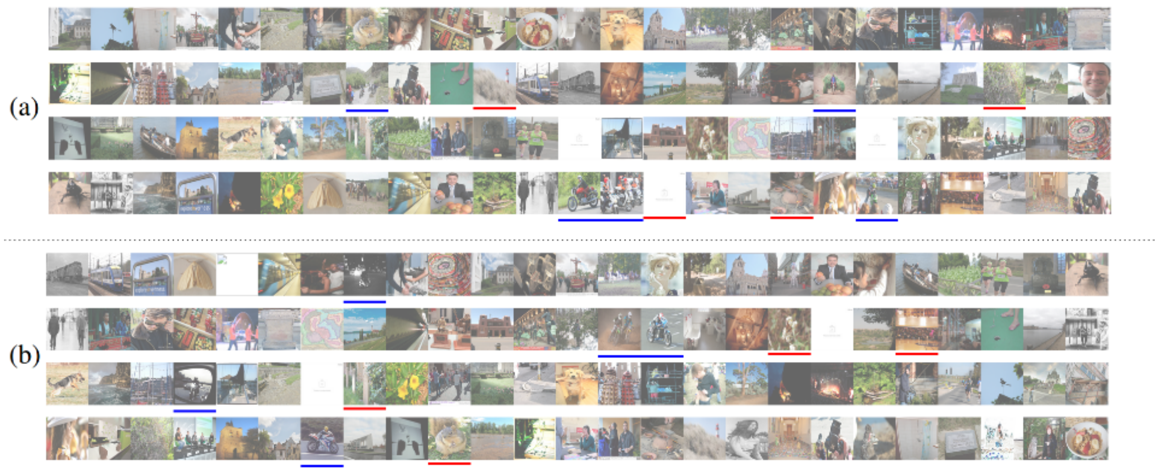


Figure 4.3: Example raw worker outputs from our interface. Each image was displayed for 100ms and workers were asked to react whenever they saw images of “a person riding a motorcycle.” Images are shown in the same order they appeared in for the worker. Positive images are shown with a blue bar below them and users’ keypresses are shown as red bars below the image to which they reacted.

is to also briefly display the last four items previously shown when the spacebar is pressed, so that workers see the one they intended and also gather an intuition for how far back the model looks. For example, in Figure 4.2(d), we show a worker pressing the spacebar on an image of a horse. We anticipate that the worker was probably delayed, and we display the last four items to acknowledge that we have recorded the keypress. We ask all workers to first complete a qualification task in which they receive feedback on how quickly we expect them to react. They pass the qualification task only if they achieve a recall of 0.6 and precision of 0.9 on a stream of 200 items with 25 positives. We measure precision as the fraction of worker reactions that were within 500ms of a positive cue.

In Figure 4.3, we show two sample outputs from our interface. Workers were shown images for 100ms each. They were asked to press the spacebar whenever they saw an image of “a person riding a motorcycle.” The images with blue bars underneath them are ground truth images of “a person riding a motorcycle.” The images with red bars show where workers reacted. The important element is that red labels are often delayed behind blue ground truth and occasionally missed entirely. Both Figures 4.3(a) and 4.3(b) have 100 images each with 5 correct images.

Because of workers’ reaction delay, the data from one worker has considerable uncertainty. We thus show the same set of items to multiple workers in different random orders and collect independent sets of keypresses. This randomization will produce a cleaner signal in aggregate and later allow us to estimate the images to which each worker intended to react.

Given the speed of the images, workers are not able to detect every single positive image. For example, the last positive image in Figure 4.3(a) and the first positive image in Figure 4.3(b) are not detected. Previous work on RSVP found a phenomenon called “attention blink” [16], in which a worker is momentarily blind to successive positive images. However, we find that even if two images of "a person riding a motorcycle" occur consecutively, workers are able to detect both and react twice (Figures 4.3(a) and 4.3(b)). If workers are forced to react in intervals of less than 400ms, though, the signal we extract is too noisy for our model to estimate the positive items.

4.3.2 Multi-Class Classification for Categorical Data

So far, we have described how rapid crowdsourcing can be used for binary verification tasks. Now we extend it to handle multi-class classification. Theoretically, all multi-class classification can be broken down into a series of binary verifications. For example, if there are N classes, we can ask N binary questions of whether an item is in each class. Given a list of items, we use our technique to classify them one class at a time. After every iteration, we remove all the positively classified items for a particular class. We use the rest of the items to detect the next class.

Assuming all the classes contain an equal number of items, the order in which we detect classes should not matter. A simple *baseline approach* would choose a class at random and attempt to detect all items for that class first. However, if the distribution of items is not equal among classes, this method would be inefficient. Consider the case where we are trying to classify items into 10 classes, and one class has 1000 items while all other classes have 10 items. In the worst case, if we classify the class with 1000 examples last, those 1000 images would go through our interface 10 times (once for every class). Instead, if we had detected the large class first, we would be able to classify those 1000 images and they would only go through our interface once. With this intuition, we propose a *class-optimized approach* that classifies the most common class of items first. We maximize the number of items we classify at every iteration, reducing the total number of binary verifications required.

4.4 Model

To translate workers' delayed and potentially erroneous actions into identifications of the positive items, we need to model their behavior. We do this by calculating the probability that a particular item is in the positive class given that the user reacted a given period after the item was displayed. By combining these probabilities across several workers with different random orders of the same images, these probabilities sum up to identify the correct items.

We use maximum likelihood estimation to predict the probability of an item being a positive example. Given a set of items $\mathcal{I} = \{I_1, \ldots, I_n\}$, we send them to W workers in a different random order for each. From each worker w, we collect a set of keypresses $\mathcal{C}^w = \{c_1^w, \ldots, c_k^w\}$ where $w \in W$ and k is the total number of keypresses from w. Our aim is to calculate the probability of a given item $P(I_i)$ being a positive example. Given that we collect keypresses from W workers:

$$P(I_i) = \sum_w P(I_i|\mathcal{C}^w)P(\mathcal{C}^w) \tag{4.1}$$

where $P(\mathcal{C}) = \prod_k P(\mathcal{C}_k)$ is the probability of a particular set of items being keypresses. We set $P(C_k)$ to be constant, asssuming that it is equally likely that a worker might react to any item. Using Bayes' rule:

$$P(I_i|\mathcal{C}^w) = \frac{P(\mathcal{C}^w|I_i)P(I_i)}{P(\mathcal{C}^w)}. \tag{4.2}$$

$P(I_i)$ models our estimate of item I_i being positive. It can be a constant, or it can be an estimate from a domain-specific machine learning algorithm [103]. For example, to calculate $P(I_i)$, if we were trying to scale up a dataset of "dog" images, we would use a small set of known "dog" images to train a binary classifier and use that to calculate $P(I_i)$ for all the unknown images. With image tasks, we use a pretrained convolutional neural network to extract image features [216] and train a linear support vector machine to calculate $P(I_i)$.

We model $P(\mathcal{C}^w|I_i)$ as a set of independent keypresses:

$$P(\mathcal{C}^w|I_i) = P(c_1^w, \ldots, c_k^w|I_i) = \prod_k P(\mathcal{C}_k^w|I_i). \tag{4.3}$$

Finally, we model each keypress as a Gaussian distribution $\mathcal{N}(\mu, \sigma)$ given a positive item. We train the mean μ and variance σ by running rapid crowdsourcing on a small set of items for which we already know the positive items. Here, the mean and variance of the distribution are modeled to estimate the delays that a worker makes when reacting to a positive item.

Intuitively, the model works by treating each keypress as creating a Gaussian "footprint" of positive probability on the images about 400ms before the keypress (Figure 4.1). The model combines these probabilities across several workers to identify the images with the highest overall probability.

Now that we have a set of probabilities for each item, we need to decide which ones should be classified as positive. We order the set of items $\mathcal{I}$ according to likelihood of being in the positive class $P(I_i)$. We then set all items above a certain threshold as positive. This threshold is a hyperparameter that can be tuned to trade off precision vs. recall.

In total, this model has two hyperparameters: (1) the threshold above which we classify images as positive and (2) the speed at which items are displayed to the user. We model both hyperparameters in a per-task (image verification, sentiment analysis, etc.) basis. For a new task, we first estimate how long it takes to label each item in the conventional setting with a small set of items. Next, we continuously reduce the time each item is displayed until we reach a point where the model is unable to achieve the same precision as the untimed case.

4.5 Calibration: Baseline Worker Reaction Time

Our technique hypothesizes that guiding workers to work quickly and make errors can lead to results that are faster yet with similar precision. We begin evaluating our technique by first studying worker

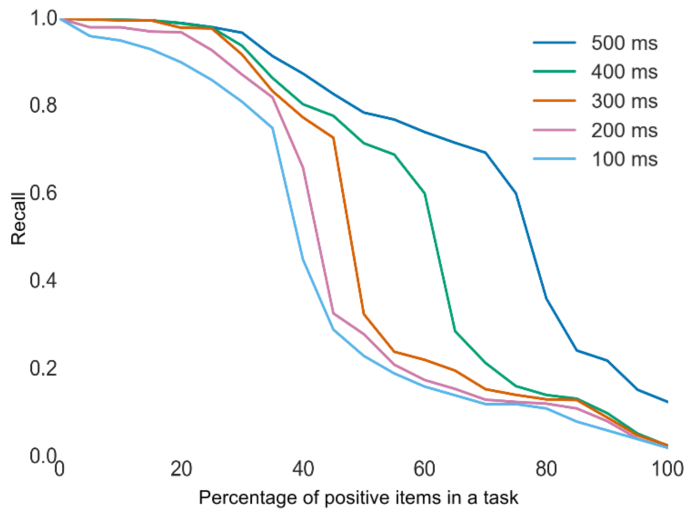


Figure 4.4: We plot the change in recall as we vary percentage of positive items in a task. We experiment at varying display speeds ranging from 100ms to 500ms. We find that recall is inversely proportional to the rate of positive stimuli and not to the percentage of positive items.

reaction times as we vary the length of time for which each item is displayed. If worker reaction times have a low variance, we accurately model them. Existing work on RSVP estimated that humans usually react about 400ms after being presented with a cue [248, 188]. Similarly, the model human processor [22] estimated that humans perceive, understand and react at least 240ms after a cue. We first measure worker reaction times, then analyze how frequently positive items can be displayed before workers are unable to react to them in time.

Method. We recruited 1,000 workers on Amazon Mechanical Turk with 96% approval rating and over 10,000 tasks submitted. Workers were asked to work on one task at a time. Each task contained a stream of 100 images of polka dot patterns of two different colors. Workers were asked to react by pressing the spacebar whenever they saw an image with polka dots of one of the two colors. Tasks could vary by two variables: the *speed* at which images were displayed and the *percentage* of the positively colored images. For a given task, we held the display speed constant. Across multiple tasks, we displayed images for 100ms to 500ms. We studied two variables: *reaction time* and *recall.* We measured the reaction time to the positive color across these speeds. To study recall (percentage of positively colored images detected by workers), we varied the ratio of positive images from 5% to 95%. We counted a keypress as a detection only if it occurred within 500ms of displaying a positively colored image.

Results. Workers' reaction times corresponded well with estimates from previous studies. Workers tend to react an average of 378ms ($\sigma = 92$ms) after seeing a positive image. This consistency is an important result for our model because it assumes that workers have a consistent reaction delay.

Task		**Conventional Approach**			**Our Technique**			**Speedup**
		Time (s)	*Precision*	*Recall*	*Time (s)*	*Precision*	*Recall*	
Image Verification	Easy	1.50	0.99	0.99	0.10	0.99	0.94	**9.00**×
	Medium	1.70	0.97	0.99	0.10	0.98	0.83	**10.20**×
	Hard	1.90	0.93	0.89	0.10	0.90	0.74	**11.40**×
	All Concepts	1.70	0.97	0.96	0.10	0.97	0.81	**10.20**×
Sentiment Analysis		4.25	0.93	0.97	0.25	0.94	0.84	**10.20**×
Word Similarity		6.23	0.89	0.94	0.60	0.88	0.86	**6.23**×
Topic Detection		14.33	0.96	0.94	2.00	0.95	0.81	**10.75**×

Table 4.1: We compare the conventional approach for binary verification tasks (image verification, sentiment analysis, word similarity and topic detection) with our technique and compute precision and recall scores. Precision scores, recall scores and speedups are calculated using 3 workers in the conventional setting. Image verification, sentiment analysis and word similarity used 5 workers using our technique, while topic detection used only 2 workers. We also show the time taken (in seconds) for 1 worker to do each task.

As expected, recall is inversely proportional to the speed at which the images are shown. A worker is more likely to miss a positive image at very fast speeds. We also find that recall decreases as we increase the percentage of positive items in the task. To measure the effects of positive frequency on recall, we record the percentage threshold at which recall begins to drop significantly at different speeds and positive frequencies. From Figure 4.4, at 100ms, we see that recall drops when the percentage of positive images is more than 35%. As we increase the time for which an item is displayed, however, we notice that the drop in recall occurs at a much higher percentage. At 500ms, the recall drops at a threshold of 85%. We thus infer that recall is inversely proportional to the *rate* of positive stimuli and not to the percentage of positive images. From these results we conclude that at faster speeds, it is important to maintain a smaller percentage of positive images, while at slower speeds, the percentage of positive images has a lesser impact on recall. Quantitatively, to maintain a recall higher than 0.7, it is necessary to limit the frequency of positive cues to one every 400ms.

4.6 Study 1: Image Verification

In this study, we deploy our technique on image verification tasks and measure its speed relative to the conventional self-paced approach. Many crowdsourcing tasks in computer vision require verifying that a particular image contains a specific class or concept. We measure precision, recall and cost (in seconds) by the conventional approach and compare against our technique.

Some visual concepts are easier to detect than others. For example, detecting an image of a "dog" is a lot easier than detecting an image of "a person riding a motorcycle" or "eating breakfast." While detecting a "dog" is a perceptual task, "a person riding a motorcycle" requires understanding of the

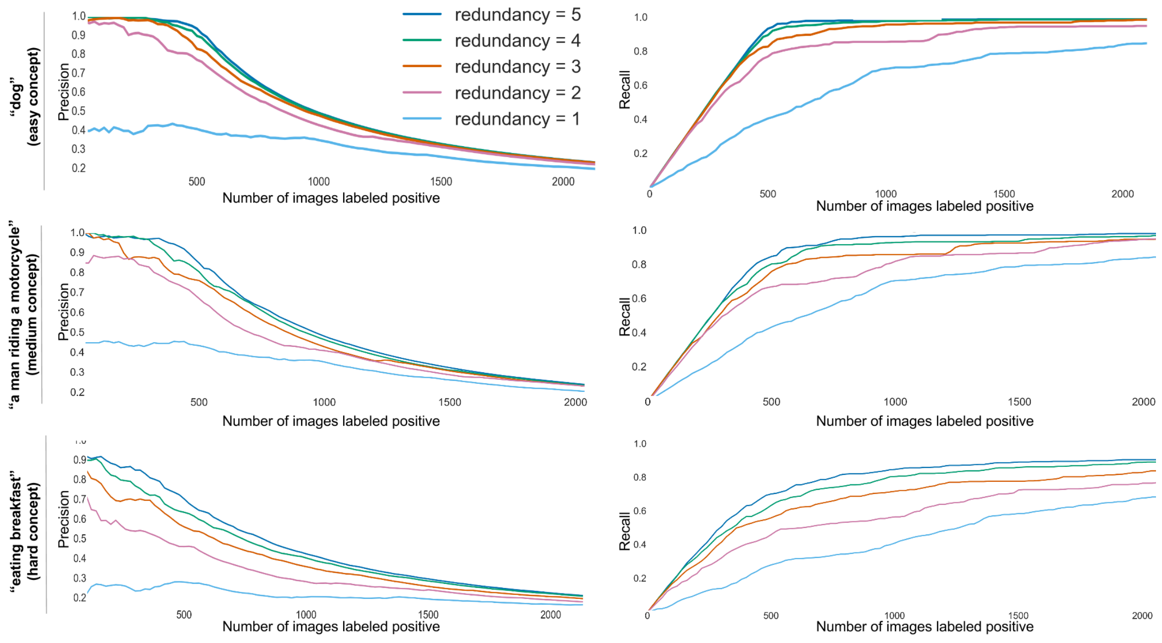


Figure 4.5: We study the precision (left) and recall (right) curves for detecting "dog" (top), "a person on a motorcycle" (middle) and "eating breakfast" (bottom) images with a redundancy ranging from 1 to 5. There are 500 ground truth positive images in each experiment. We find that our technique works for simple as well as hard concepts.

interaction between the person and the motorcycle. Similarly, "eating breakfast" requires workers to fuse concepts of people eating a variety foods like eggs, cereal or pancakes. We test our technique on detecting three concepts: "dog" (easy concept), "a person riding a motorcycle" (medium concept) and "eating breakfast" (hard concept). In this study, we compare how workers fare on each of these three levels of concepts.

Method. In this study, we compare the conventional approach with our technique on three (easy, medium and hard) concepts. We evaluate each of these comparisons using precision scores, recall scores and the speedup achieved. To test each of the three concepts, we labeled 10,000 images, where each concept had 500 examples. We divided the 10,000 images into streams of 100 images for each task. We paid workers $0.17 to label a stream of 100 images (resulting in a wage of $6 per hour [205]). We hired over 1,000 workers for this study satisfying the same qualifications as the calibration task.

The conventional method of collecting binary labels is to present a crowd worker with a set of items. The worker proceeds to label each item, one at a time. Most datasets employ multiple workers to label each task because majority voting [221] has been shown to improve the quality of crowd annotations. These datasets usually use a redundancy of 3 to 5 workers [213]. In all our experiments, we used a redundancy of 3 workers as our baseline.

When launching tasks using our technique, we tuned the image display speed to 100ms. We used a redundancy of 5 workers when measuring precision and recall scores. To calculate *speedup*, we compare the total worker time taken by all the 5 workers using our technique with the total worker time taken by the 3 workers using the conventional method. Additionally, we vary redundancy on all the concepts to from 1 to 10 workers to see its effects on precision and recall.

Results. Self-paced workers take 1.70s on average to label each image with a concept in the conventional approach (Table 4.1). They are quicker at labeling the easy concept (1.50s per worker) while taking longer on the medium (1.70s) and hard (1.90s) concepts.

Using our technique, even with a redundancy of 5 workers, we achieve a speedup of 10.20× across all concepts. We achieve *order of magnitude* speedups of 9.00×, 10.20× and 11.40× on the easy, medium and hard concepts. Overall, across all concepts, the precision and recall achieved by our technique is 0.97 and 0.81. Meanwhile the precision and recall of the conventional method is 0.97 and 0.96. We thus achieve the same precision as the conventional method. As expected, recall is lower because workers are not able to detect every single true positive example. As argued previously, lower recall can be an acceptable tradeoff when it is easy to find more unlabeled images.

Now, let's compare precision and recall scores between the three concepts. We show precision and recall scores in Figure 4.5 for the three concepts. Workers perform slightly better at finding "dog" images and find it the most difficult to detect the more challenging "eating breakfast" concept. With a redundancy of 5, the three concepts achieve a precision of 0.99, 0.98 and 0.90 respectively at a recall of 0.94, 0.83 and 0.74 (Table 4.1). The precision for these three concepts are identical to the conventional approach, while the recall scores are slightly lower. The recall for a more difficult cognitive concept ("eating breakfast") is much lower, at 0.74, than for the other two concepts. More complex concepts usually tend to have a lot of contextual variance. For example, "eating breakfast" might include a person eating a "banana," a "bowl of cereal," "waffles" or "eggs." We find that while some workers react to one variety of the concept (e.g., "bowl of cereal"), others react to another variety (e.g., "eggs").

When we increase the redundancy of workers to 10 (Figure 4.6), our model is able to better approximate the positive images. We see diminishing increases in both recall and precision as redundancy increases. At a redundancy of 10, we increase recall to the same amount as the conventional approach (0.96), while maintaining a high precision (0.99) and still achieving a speedup of 5.1×.

We conclude from this study that our technique (with a redundancy of 5) can speed up image verification with easy, medium and hard concepts by an order of magnitude while still maintaining high precision. We also show that recall can be compensated by increasing redundancy.

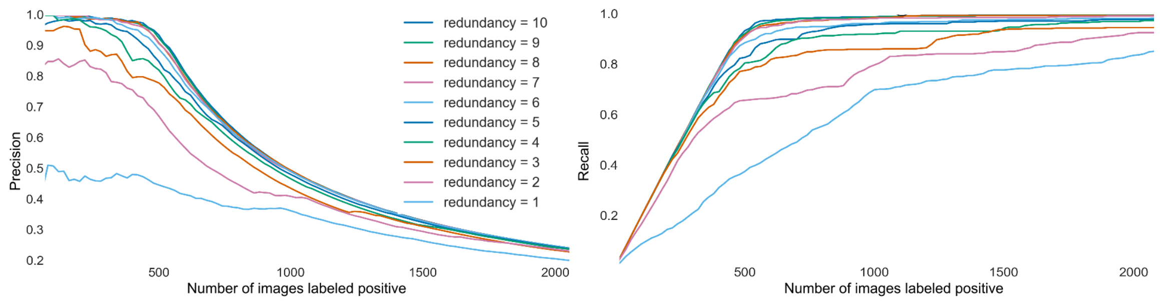


Figure 4.6: We study the effects of redundancy on recall by plotting precision and recall curves for detecting "a person on a motorcycle" images with a redundancy ranging from 1 to 10. We see diminishing increases in precision and recall as we increase redundancy. We manage to achieve the same precision and recall scores as the conventional approach with a redundancy of 10 while still achieving a speedup of 5×.

4.7 Study 2: Non-Visual Tasks

So far, we have shown that rapid crowdsourcing can be used to collect image verification labels. We next test the technique on a variety of other common crowdsourcing tasks: sentiment analysis [170], word similarity [221] and topic detection [133].

Method. In this study, we measure precision, recall and speedup achieved by our technique over the conventional approach. To determine the stream speed for each task, we followed the prescribed method of running trials and speeding up the stream until the model starts losing precision. For sentiment analysis, workers were shown a stream of tweets and asked to react whenever they saw a positive tweet. We displayed tweets at 250ms with a redundancy of 5 workers. For word similarity, workers were shown a word (e.g., "lad") for which we wanted synonyms. They were then rapidly shown other words at 600ms and asked to react if they see a synonym (e.g., "boy"). Finally, for topic detection, we presented workers with a topic like "housing" or "gas" and presented articles of an average length of 105 words at a speed of 2s per article. They reacted whenever they saw an article containing the topic we were looking for. For all three of these tasks, we compare precision, recall and speed against the self-paced conventional approach with a redundancy of 3 workers. Every task, for both the conventional approach and our technique, contained 100 items.

To measure the cognitive load on workers for labeling so many items at once, we ran the widely-used NASA Task Load Index (TLX) [36] on all tasks, including image verification. TLX measures the perceived workload of a task. We ran the survey on 100 workers who used the conventional approach and 100 workers who used our technique across all tasks.

Results. We present our results in Table 4.1 and Figure 4.7. For sentiment analysis, we find that workers in the conventional approach classify tweets in 4.25s. So, with a redundancy of 3 workers, the conventional approach would take 12.75s with a precision of 0.93. Using our method and a

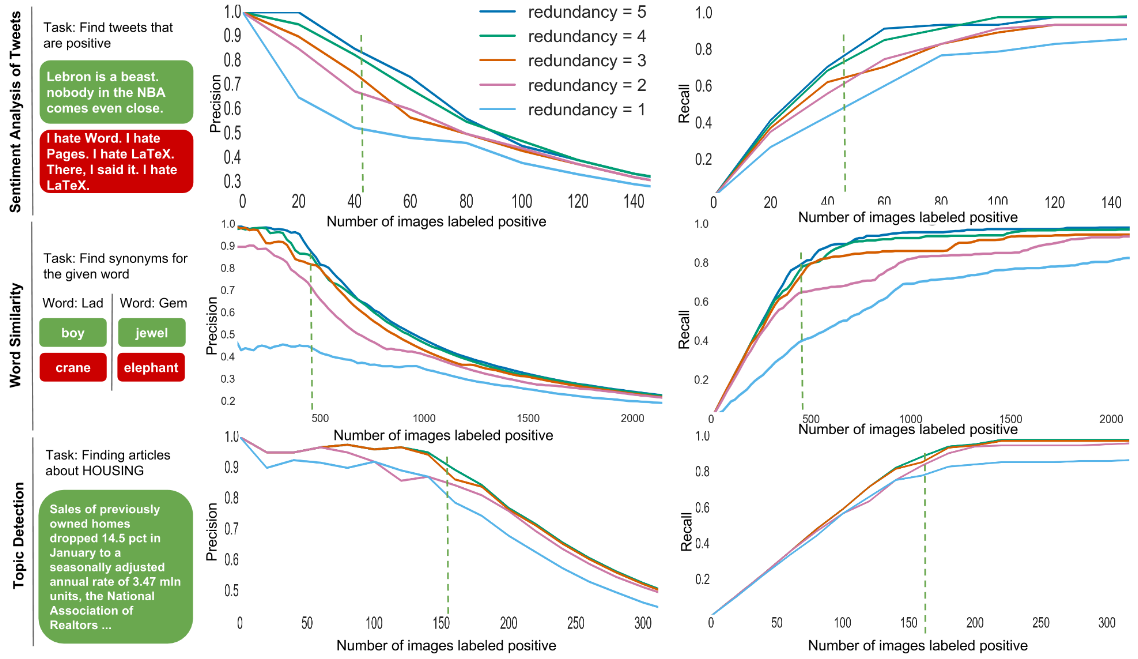


Figure 4.7: Precision (left) and recall (right) curves for sentiment analysis (top), word similarity (middle) and topic detection (bottom) images with a redundancy ranging from 1 to 5. Vertical lines indicate the number of ground truth positive examples.

redundancy of 5 workers, we complete the task in 1250ms (250ms per worker per item) and 0.94 precision. Therefore, our technique achieves a speedup of 10.2×.

Likewise, for word similarity, workers take around 6.23s to complete the conventional task, while our technique succeeds at 600ms. We manage to capture a comparable precision of 0.88 using 5 workers against a precision of 0.89 in the conventional method with 3 workers. Since finding synonyms is a higher-level cognitive task, workers take longer to do word similarity tasks than image verification and sentiment analysis tasks. We manage a speedup of 6.23×.

Finally, for topic detection, workers spend significant time analyzing articles in the conventional setting (14.33s on average). With 3 workers, the conventional approach takes 43s. In comparison, our technique delegates 2s for each article. With a redundancy of only 2 workers, we achieve a precision of 0.95, similar to the 0.96 achieved by the conventional approach. The total worker time to label one article using our technique is 4s, a speedup of 10.75×.

The mean TLX workload for the control condition was 58.5 ($\sigma = 9.3$), and 62.4 ($\sigma = 18.5$) for our technique. Unexpectedly, the difference between conditions was not significant ($t(99) = -0.53, p = 0.59$). The temporal demand scale item appeared to be elevated for our technique (61.1 vs. 70.0), but this difference was not significant ($t(99) = -0.76, p = 0.45$). We conclude that our technique can be used to scale crowdsourcing on a variety of tasks without statistically increasing worker workload.

4.8 Study 3: Multi-class Classification

In this study, we extend our technique from binary to multi-class classification to capture an even larger set of crowdsourcing tasks. We use our technique to create a dataset where each image is classified into one category ("people," "dog," "horse," "cat," etc.). We compare our technique with a conventional technique [40] that collects binary labels for each image for every single possible class.

Method. Our aim is to classify a dataset of 2,000 images with 10 categories where each category contains between 100 to 250 examples. We compared three methods of multi-class classification: (1) a *naive approach* that collected 10 binary labels (one for each class) for each image, (2) a *baseline approach* that used our interface and classified images one class (chosen randomly) at a time, and (3) a *class-optimized approach* that used our interface to classify images starting from the class with the most examples. When using our interface, we broke tasks into streams of 100 images displayed for 100ms each. We used a redundancy of 3 workers for the conventional interface and 5 workers for our interface. We calculated the precision and recall scores across each of these three methods as well as the cost (in seconds) of each method.

Results. (1) In the *naive approach*, we need to collect 20,000 binary labels that take 1.7s each. With 5 workers, this takes 102,000s ($170 at a wage rate of $6/hr) with an average precision of 0.99 and recall of 0.95. (2) Using the *baseline approach*, it takes 12,342s ($20.57) with an average precision of 0.98 and recall of 0.83. This shows that the baseline approach achieves a speedup of 8.26× when compared with the naive approach. (3) Finally, the *class-optimized approach* is able to detect the most common class first and hence reduces the number of times an image is sent through our interface. It takes 11,700s ($19.50) with an average precision of 0.98 and recall of 0.83. The class-optimized approach achieves a speedup of 8.7× when compared to the naive approach. While the speedup between the baseline and the class-optimized methods is small, it would be increased on a larger dataset with more classes.

4.9 Application: Building ImageNet

Our method can be combined with existing techniques [41, 226, 173, 12] that optimize binary verification and multi-class classification by preprocessing data or using active learning. One such method [41] annotated ImageNet (a popular large dataset for image classification) effectively with a useful insight: they realized that its classes could be grouped together into higher semantic concepts. For example, "dog," "rabbit" and "cat" could be grouped into the concept "animal." By utilizing the hierarchy of labels that is specific to this task, they were able to preprocess and reduce the number of labels needed to classify all images. As a case study, we combine our technique with their insight and evaluate the speedup in collecting a subset of ImageNet.

Method. We focused on a subset of the dataset with 20,000 images and classified them into 200 classes. We conducted this case study by comparing three ways of collecting labels: (1) The naive approach asked 200 binary questions for each image in the subset, where each question asked if the image belonged to one of the 200 classes. We used a redundancy of 3 workers for this task. (2) The optimal-labeling method used the insight to reduce the number of labels by utilizing the hierarchy of image classes. (3) The combined approach used our technique for multi-class classification combined with the hierarchy insight to reduce the number of labels collected. We used a redundancy of 5 workers for this technique with tasks of 100 images displayed at 250ms.

Results. (1) Using the naive approach, this would result in asking 4 million binary verification questions. Given that each binary label takes 1.7s (Table 4.1), we estimate that the total time to label the entire dataset would take 6.8 million seconds ($11,333 at a wage rate of $6/hr). (2) The optimal-labeling method is estimated to take 1.13 million seconds ($1,888) [41]. (3) Combining the hierarchical questions with our interface, we annotate the subset in 136,800s ($228). We achieve a precision of 0.97 with a recall of 0.82. By combining our 8× speedup with the 6× speedup from intelligent question selection, we achieve a 50× speedup in total.

4.10 Discussion

We focused our technique on positively identifying concepts. We then also test its effectiveness at classifying the absence of a concept. Instead of asking workers to react when they see a “dog,” if we ask them to react when they do *not* see a “dog,” our technique performs poorly. At 100ms, we find that workers achieve a recall of only 0.31, which is much lower than a recall of 0.94 when detecting the presence of “dog”s. To improve recall to 0.90, we must slow down the feed to 500ms. Our technique achieves a speedup of 2× with this speed. We conclude that our technique performs poorly for anomaly detection tasks, where the presence of a concept is common but its absence, an anomaly, is rare. More generally, this exercise suggests that some cognitive tasks are less robust to rapid judgments. Preattentive processing can help us find “dog”s, but ensuring that there is no “dog” requires a linear scan of the entire image.

To better understand the active mechanism behind our technique, we turn to concept typicality. A recent study [89] used fMRIs to measure humans’ recognition speed for different object categories, finding that images of most typical examplars from a class were recognized faster than the least typical categories. They calculated typicality scores for a set of image classes based on how quickly humans recognized them. In our image verification task, 72% of false negatives were also atypical. Not detecting atypical images might lead to the curation of image datasets that are biased towards more common categories. For example, when curating a dataset of dogs, our technique would be more likely to find usual breeds like “dalmatians” and “labradors” and miss rare breeds like “romagnolos” and “otterhounds.” More generally, this approach may amplify biases and minimize clarity on edge cases. Slowing down the feed reduces atypical false negatives, resulting in a smaller speedup but with a higher recall for atypical images.

4.11 Conclusion

We have suggested that crowdsourcing can speed up labeling by encouraging a small amount of error rather than forcing workers to avoid it. We introduce a rapid slideshow interface where items are shown too quickly for workers to get all items correct. We algorithmically model worker errors and recover their intended labels. This interface can be used for binary verification tasks like image verification, sentiment analysis, word similarity and topic detection, achieving speedups of $10.2\times$, $10.2\times$, $6.23\times$ and $10.75\times$ respectively. It can also extend to multi-class classification and achieve a speedup of $8.26\times$. Our approach is only one possible interface instantiation of the concept of encouraging some error; we suggest that future work may investigate many others. Speeding up crowdsourcing enables us to build larger datasets to empower scientific insights and industry practice. For many labeling goals, this technique can be used to construct datasets that are an order of magnitude larger without increasing cost.

Chapter 5

Visual Genome Experiments

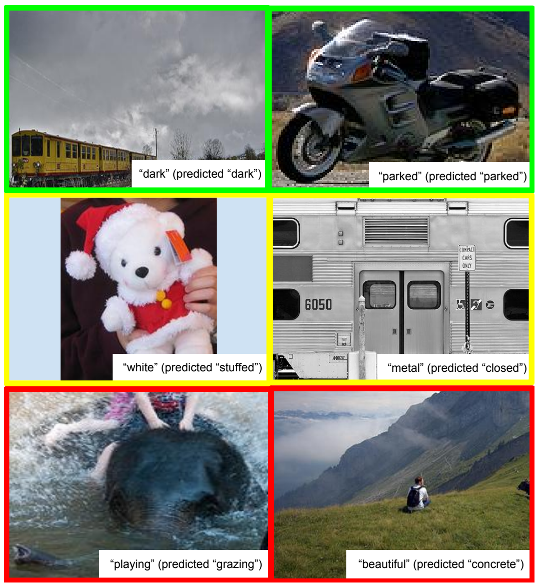
5.1 Introduction

Thus far, we have presented the Visual Genome dataset and analyzed its individual components. With such rich information provided, numerous perceptual and cognitive tasks can be tackled. In this chapter, we aim to provide baseline experimental results using components of Visual Genome that have not been extensively studied. Object detection is already a well-studied problem [46, 71, 211, 70, 191]. We therefore focus on the remaining components, i.e. *attributes*, *relationships*, *region descriptions*, and *question answer pairs*.

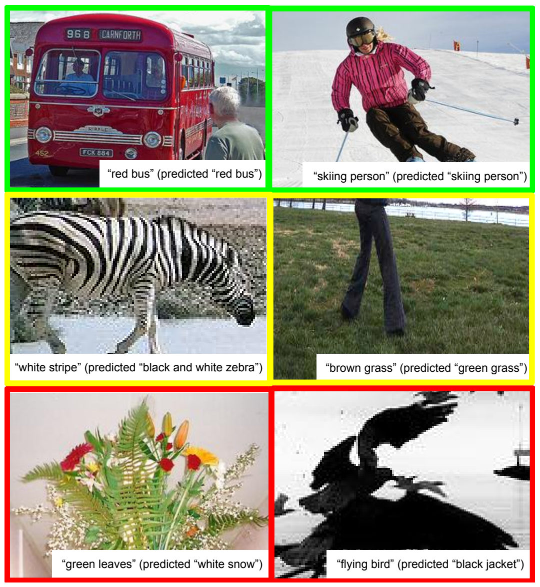
In Chapter 5.2, we present results for two experiments on attribute prediction. In the first, we treat attributes independently from objects and train a classifier for each attribute, i.e. a classifier for `red` or a classifier for `old`, as in [147, 240, 55, 50, 100]. In the second experiment, we learn object and attribute classifiers *jointly* and predict object-attribute pairs (e.g. predicting that an `apple` is `red`), as in [203].

In Chapter 5.3, we present two experiments on relationship prediction. In the first, we aim to predict the predicate between two objects, e.g. predicting the predicate `kicking` or `wearing` between two objects. This experiment is synonymous with existing work in action recognition [79, 187]. In another experiment, we study relationships by classifying jointly the objects and the predicate (e.g. predicting *kicking*(*man*, *ball*)); we show that this is a very difficult task due to the high variability in the appearance of a relationship (e.g. the `ball` might be on the ground or in mid-air above the `man`). These experiment are generalizations of tasks that study spatial relationships between objects and ones that jointly reason about the interaction of humans with objects [257, 183].

In Chapter 5.4 we present results for region captioning. This task is closely related to image captioning [25]; however, results from the two are not directly comparable, as region descriptions are short, incomplete sentences. We train one of the top 16 state-of-the-art image caption generator [106] on (1) our dataset to generate region descriptions and on (2) Flickr30K [262] to generate sentence


"dark" (predicted "dark")
"parked" (predicted "parked")
"white" (predicted "stuffed")
"metal" (predicted "closed")
"playing" (predicted "grazing")
"beautiful" (predicted "concrete")

(a)


"red bus" (predicted "red bus")
"skiing person" (predicted "skiing person")
"white stripe" (predicted "black and white zebra")
"brown grass" (predicted "green grass")
"green leaves" (predicted "white snow")
"flying bird" (predicted "black jacket")

(b)

Figure 5.1: (a) Example predictions from the attribute prediction experiment. Attributes in the first row are predicted correctly, those in the second row differ from the ground truth but still correctly classify an attribute in the image, and those in the third row are classified incorrectly. The model tends to associate objects with attributes (e.g. `elephant` with `grazing`). (b) Example predictions from the joint object-attribute prediction experiment.

descriptions. To compare results between the two training approaches, we use simple templates to convert region descriptions into complete sentences. For a more robust evaluation, we validate the descriptions we generate using human judgment.

Finally, in Chapter 5.5, we experiment on visual question answering, i.e. given an image and a question, we attempt to provide an answer for the question. We report results on the retrieval of the correct answer from a list of existing answers.

5.2 Attribute Prediction

Attributes are becoming increasingly important in the field of computer vision, as they offer higher-level semantic cues for various problems and lead to a deeper understanding of images. We can express a wide variety of properties through attributes, such as form (`sliced`), function (`decorative`), sentiment (`angry`), and even intention (`helping`). Distinguishing between similar objects [93] leads to finer-grained classification, while describing a previously unseen class through attributes shared with known classes can enable "zero-shot" learning [50, 125]. Visual Genome is the largest dataset of attributes, with 18 attributes per image for a total of 1.8 million attributes.

Setup. For both experiments, we focus on the 100 most common attributes in our dataset. We only use objects that occur at least 100 times and are associated with one of the 100 attributes in at least one image. For both experiments, we follow a similar data pre-processing pipeline. First, we lowercase, lemmatize, and strip excess whitespace from all attributes. Since the number of examples per attribute class varies, we randomly sample 500 attributes from each category (if fewer than 500 are in the class, we take all of them).

We end up with around $50,000$ attribute instances and $43,000$ object-attribute pair instances in total. We use 80% of the images for training and 10% each for validation and testing. Because each image has about the same number of examples, this results in an approximately 80%-10%-10% split over the attributes themselves. The input data for this experiment is the cropped bounding box of the object associated with each attribute.

We train an attribute predictor by using features learned from a convolutional neural network. Specifically, we fine-tune a 16-layer VGG network [217] for both of these experiments using the $50,000$ attribute and $43,000$ object-attribute pair instances respectively. We modify the network so that the learning rate of the final fully-connected layer is 10 times that of the other layers, as this improves convergence time. We use a base learning rate of 0.001, which we scale by 0.1 every 200 iterations, and momentum and weight decays of 0.9 and 0.0005 respectively. We use the fine-tuned features from the network and train 100 individual SVMs [82] to predict each attribute. We output multiple attributes for each bounding box input. For the second experiment, we also output the object class.

Results. Table 5.1 shows results for both experiments. For the first experiment on attribute prediction, we converge after around 700 iterations with 18.97% top-one accuracy and 43.11% top-five accuracy. Thus, attributes (like objects) are visually distinguishable from each other. For the second experiment where we also predict the object class, we converge after around 400 iterations with 43.17% top-one accuracy and 71.97% top-five accuracy. Predicting objects jointly with attributes increases the top-one accuracy from 18.97% to 43.17%. This implies that some attributes occur exclusively with a small number of objects. Additionally, by jointly learning attributes with objects, we increase the inter-class variance, making the classification process an easier task.

Figure 5.1 (a) shows example predictions for the first attribute prediction experiment. In general, the model is good at associating objects with their most salient attributes, for example, `animal` with `stuffed` and `elephant` with `grazing`. However, there is some difference between the user-provided result and the correct ground truth, so the model incorrectly classifies some correct predictions. For example, the `white` stuffed animal is correct but evaluated as incorrect.

Figure 5.1 (b) shows example predictions for the second experiment in which we also predict the object. While the results in the second row might be considered correct, to keep a consistent evaluation, we mark them as incorrect. For example, the predicted "green grass" might be considered subjectively correct even though it is annotated as "brown grass". For cases where the objects are

	Top-1 Accuracy	Top-5 Accuracy
Attribute	18.97%	43.11%
Object-Attribute	43.17%	71.97%

Table 5.1: (First row) Results for the attribute prediction task where we only predict attributes for a given image crop. (Second row) Attribute-object prediction experiment where we predict both the attributes as well as the object from a given crop of the image.

not clearly visible but are abstract outlines, our model is unable to predict attributes or objects accurately. For example, it thinks that the "flying bird" is actually a "black jacket".

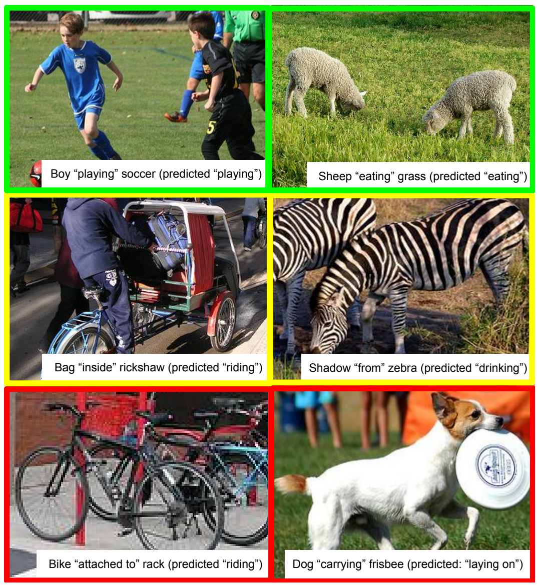
The attribute clique graphs in Chapter 2.2.4 clearly show that learning attributes can help us identify types of objects. This experiment strengthens that insight. We learn that studying attributes together with objects can improve attribute prediction.

5.3 Relationship Prediction

While objects are the core building blocks of an image, relationships put them in context. These relationships help distinguish between images that contain the same objects but have different holistic interpretations. For example, an image of "a man riding a bike" and "a man falling off a bike" both contain `man` and `bike`, but the relationship (`riding` vs. `falling_off`) changes how we perceive both situations. Visual Genome is the largest known dataset of relationships, with a total of 1.8 million relationships and an average of 18 relationships per image.

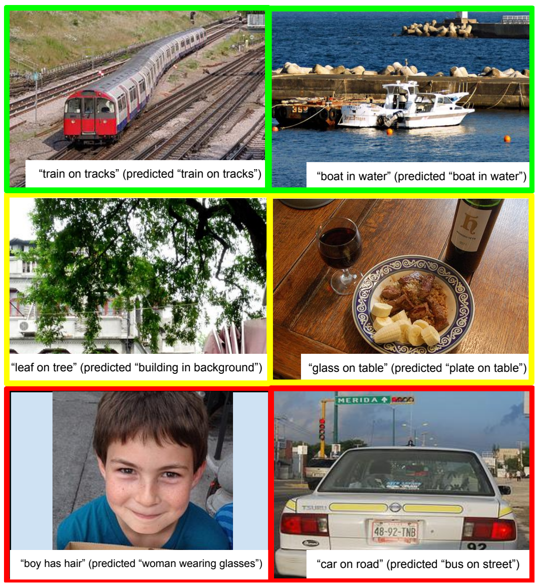
Setup. The setups of both experiments are similar to those of the experiments we performed on attributes. We again focus on the top 100 most frequent relationships. We lowercase, lemmatize, and strip excess whitespace from all relationships. We end up with around $34,000$ relationships and $27,000$ subject-relationship-object triples for training, validation, and testing. The input data to the experiment is the image region containing the union of the bounding boxes of the subject and object (essentially, the bounding box containing the two object boxes). We fine-tune a 16-layer VGG network [217] with the same learning rates mentioned in Chapter 5.2.

Results. Overall, we find that relationships are not visually distinct enough for our discriminative model to learn effectively. Table 5.2 shows results for both experiments. For relationship classification, we converge after around 800 iterations with 8.74% top-one accuracy and 29.69% top-five accuracy. Unlike attribute prediction, the accuracy results for relationships are much lower because of the high intra-class variability of most relationships. For the second experiment jointly predicting the relationship and its two object classes, we converge after around 450 iterations with 25.83% top-one accuracy and 65.57% top-five accuracy. We notice that object classification aids relationship prediction. Some relationships occur with some objects and never others; for example, the



Boy "playing" soccer (predicted "playing")
Sheep "eating" grass (predicted "eating")
Bag "inside" rickshaw (predicted "riding")
Shadow "from" zebra (predicted "drinking")
Bike "attached to" rack (predicted "riding")
Dog "carrying" frisbee (predicted: "laying on")

(a)



"train on tracks" (predicted "train on tracks")
"boat in water" (predicted "boat in water")
"leaf on tree" (predicted "building in background")
"glass on table" (predicted "plate on table")
"boy has hair" (predicted "woman wearing glasses")
"car on road" (predicted "bus on street")

(b)

Figure 5.2: (a) Example predictions from the relationship prediction experiment. Relationships in the first row are predicted correctly, those in the second row differ from the ground truth but still correctly classify a relationship in the image, and those in the third row are classified incorrectly. The model learns to associate animals leaning towards the ground as `eating` or `drinking` and bikes with `riding`. (b) Example predictions from the relationship-objects prediction experiment. The figure is organized in the same way as Figure (a). The model is able to predict the salient features of the image but fails to distinguish between different objects (e.g. `boy` and `woman` and `car` and `bus` in the bottom row).

relationship `drive` only occurs with the object `person` and never with any other objects (`dog`, `chair`, etc.).

Figure 5.2 (a) shows example predictions for the relationship classification experiment. In general, the model associates object categories with certain relationships (e.g. animals with `eating` or `drinking`, bikes with `riding`, and kids with `playing`).

Figure 5.2 (b), structured as in Figure 5.2 (a), shows example predictions for the joint prediction of relationships with its objects. The model is able to predict the salient features of the image (e.g. "boat in water") but fails to distinguish between different objects (e.g. `boy` vs. `woman` and `car` vs. `bus` in the bottom row).

	Top-1 Accuracy	Top-5 Accuracy
Relationship	8.74%	26.69%
Sub./Rel./Obj.	25.83%	65.57%

Table 5.2: Results for relationship classification (first row) and joint classification (second row) experiments.

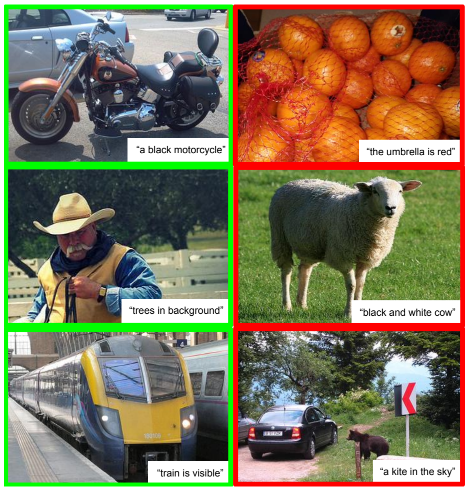


Figure 5.3: Example predictions from the region description generation experiment. Regions in the first column (left) accurately describe the region, and those in the second column (right) are incorrect and unrelated to the corresponding region.

5.4 Generating Region Descriptions

Generating sentence descriptions of images has gained popularity as a task in computer vision [108, 150, 106, 242]; however, current state-of-the-art models fail to describe all the different events captured in an image and instead provide only a high-level summary of the image. In this section, we test how well state-of-the-art models can caption the details of images. For both experiments, we use the NeuralTalk model [106], since it not only provides state-of-the-art results but also is shown to be robust enough for predicting short descriptions. We train NeuralTalk on the Visual Genome dataset for region descriptions and on Flickr30K [262] for full sentence descriptions. As a model trained on other datasets would generate complete sentences and would not be comparable [25] to our region descriptions, we convert all region descriptions generated by our model into complete sentences using predefined templates [87].

Setup. For training, we begin by preprocessing region descriptions; we remove all non-alphanumeric characters and lowercase and strip excess whitespace from them. We have $4,158,841$ region descriptions in total. We end up with $3,150,000$ region descriptions for training – $504,420$ each for validation and testing. Note that we ensure descriptions of regions from the same image are exclusively in the training, validation, or testing set. We feed the bounding boxes of the regions through the pretrained VGG 16-layer network [217] to get the 4096-dimensional feature vectors of each region. We then use the NeuralTalk [106] model to train a long short-term memory (LSTM) network [84] to generate descriptions of regions. We use a learning rate of 0.001 trained with rmsprop [38]. The model converges after four days.

For testing, we crop the ground-truth region bounding boxes of images and extract their 4096-dimensional 16-layer VGG network [217] features. We then feed these vectors through the pretrained NeuralTalk model to get predictions for region descriptions.

Results. Table 5.3 shows the results for the experiment. We calculate BLEU, CIDEr, and METEOR scores [25] between the generated descriptions and their ground-truth descriptions. In all cases, the model trained on VisualGenome performs better. Moreover, we asked crowd workers to evaluate whether a generated description was correct—we got 1.6% and 43.03% for models trained on Flickr30K and on Visual Genome, respectively. The large increase in accuracy when the model trained on our data is due to the specificity of our dataset. Our region descriptions are shorter and cover a smaller image area. In comparison, the Flickr30K data are generic descriptions of entire images with multiple events happening in different regions of the image. The model trained on our data is able to make predictions that are more likely to concentrate on the specific part of the image it is looking at, instead of generating a summary description. The objectively low accuracy in both cases illustrates that current models are unable to reason about complex images.

Figure 5.3 shows examples of regions and their predicted descriptions. Since many examples have short descriptions, the predicted descriptions are also short as expected; however, this causes the model to fail to produce more descriptive phrases for regions with multiple objects or with distinctive objects (i.e. objects with many attributes). While we use templates to convert region descriptions into sentences, future work can explore smarter approaches to combine region descriptions and generate a paragraph connecting all the regions into one coherent description.

5.5 Question Answering

Visual Genome is currently the largest dataset of visual question answers with 1.7 million question and answer pairs. Each of our $108,249$ images contains an average of 17 question answer pairs. Answering questions requires a deeper understanding of an image than generic image captioning. Question answering can involve fine-grained recognition (e.g. "What is the breed of the dog?"),

	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr	METEOR	Human
Flickr8K	0.09	0.01	0.002	0.0004	0.05	0.04	1.6%
VG	0.17	0.05	0.02	0.01	0.30	0.09	43.03%

Table 5.3: Results for the region description generation experiment. Scores in the first row are for the region descriptions generated from the NeuralTalk model trained on Flickr8K, and those in the second row are for those generated by the model trained on Visual Genome data. BLEU, CIDEr, and METEOR scores all compare the predicted description to a ground truth in different ways.

	top-100	top-500	top-1000
What	0.420	0.602	0.672
Where	0.096	0.324	0.418
When	0.714	0.809	0.834
Who	0.355	0.493	0.605
Why	0.034	0.118	0.187
How	0.780	0.827	0.846
Overall	0.411	0.573	0.641

Table 5.4: Baseline QA performances (in accuracy).

object detection (e.g. "Where is the kite in the image?"), activity recognition (e.g. "What is this man doing?"), knowledge base reasoning (e.g. "Is this glass full?"), and common-sense reasoning (e.g. "What street will we be on if we turn right?").

By leveraging the detailed annotations in the scene graphs in Visual Genome, we envision building smart models that can answer a myriad of visual questions. While we encourage the construction of smart models, in this chapter, we provide some baseline metrics to help others compare their models.

Setup. We split the QA pairs into a training set (60%) and a test set (40%). We ensure that all images are exclusive to either the training set or the test set. We implement a simple baseline model that relies on answer frequency. The model counts the top k most frequent answers (similar to the ImageNet challenge [197]) in the training set as the predictions for all the test questions, where $k = 100$, 500, and 1000. We let a model make k different predictions. We say the model is correct on a QA if one of the k predictions matches exactly with the ground-truth answer. We report the accuracy over all test questions. This evaluation method works well when the answers are short, especially for single-word answers. However, it causes problems when the answers are long phrases and sentences. Other evaluation methods require word ontologies [145], multiple choices [3, 263], or human judges [66].

Results. Table 5.4 shows the performance of the open-ended visual question answering task. These baseline results imply the long-tail distribution of the answers. Long-tail distribution is common in existing QA datasets as well [3, 145]. The top 100, 500, and 1000 most frequent answers only cover 41.1%, 57.3%, and 64.1% of the correct answers. In comparison, the corresponding sets of frequent answers in VQA [3] cover 63%, 75%, and 80% of the test set answers. The "where" and "why" questions, which tend to involve spatial and common sense reasoning, tend to have more diverse answers and hence perform poorly, with performances of 0.096% and 0.024% top-100 respectively. The top 1000 frequent answers cover only 41.8% and 18.7% of the correct answers from these two question types respectively.

Chapter 6

Visual Relationship Extraction

6.1 Introduction

While objects are the core building blocks of an image, it is often the relationships between objects that determine the holistic interpretation. For example, an image with a `person` and a `bicycle` might involve the man riding, pushing, or even falling off the bicycle (Figure 6.1). Understanding this diversity of relationships is central to accurate image retrieval and to a richer semantic understanding of our visual world.

Visual relationships are a pair of localized objects connected via a predicate (Figure 6.2). We denote relationships as *predicate*(*object*$_1$, *object*$_2$) [1]. Visual relationship detection involves detecting and localizing pairs of objects in an image and also classifying the predicate or interaction between each pair (Figure 6.2). While it poses similar challenges as object detection [46], one critical difference is that the size of the semantic space of possible relationships is much larger than that of objects. Since relationships are composed of two objects, there is a greater skew of rare relationships as object co-occurrence is infrequent in images. So, a fundamental challenge in visual relationship detection is learning from very few examples.

Visual Phrases [203] studied visual relationship detection using a small set of 13 common relationships. Their model requires enough training examples for every possible *predicate*(*object*$_1$, *object*$_2$) combination, which is difficult to collect owing to the infrequency of relationships. If we have N objects and K predicates, Visual Phrases [203] would need to train $\mathcal{O}(N^2K)$ unique detectors separately. We use the insight that while relationships (*e.g.*"person jumping over a fire hydrant") might occur rarely in images, its objects (*e.g.*`person` and `fire hydrant`) and predicate (*e.g.*`jumping over`) appear more frequently. We propose a **visual appearance module** that learns the appearance of objects and predicates and fuses them together to jointly predict relationships. We show

[1]In natural language processing [268, 77, 37, 223], relationships are defined as *predicate*(*subject*, *object*). In this chapter, we define them as *predicate*(*object*$_1$, *object*$_2$) for simplicity.

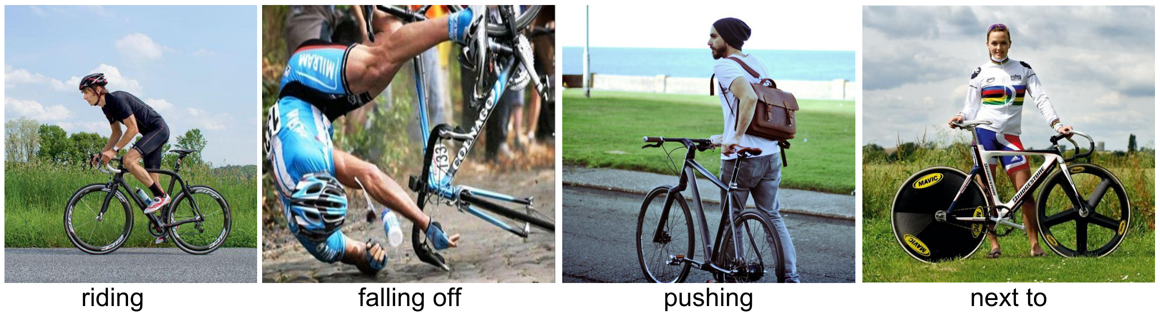


Figure 6.1: Images of a `person` and a `bicycle`. Even though they contain the same objects, it is the relationship between the objects that determine the holistic interpretation of the image.

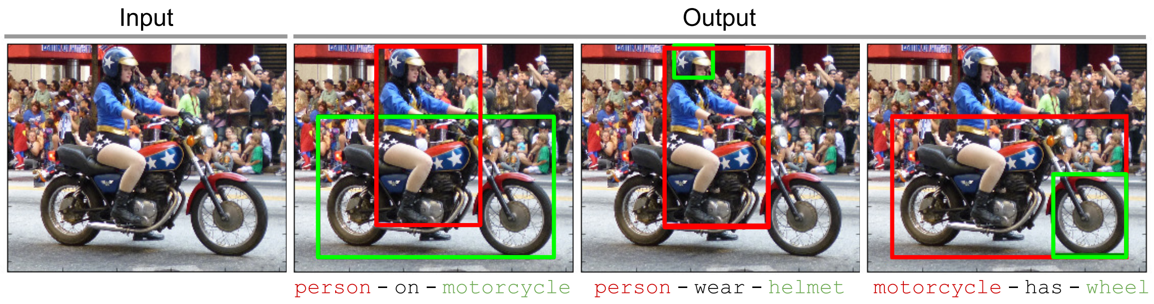


Figure 6.2: Visual Relationship Detection: Given an image as input, we detect multiple relationships in the form of *relationship*(*object*$_1$, *object*$_2$). Both the objects are localized in the image as bounding boxes. In this example, we detect the following relationships: *on*(*person*, *motorcycle*), *wear*(*person*, *helmet*) and *has*(*motorcycle*, *wheel*).

that our model only needs $\mathcal{O}(N + K)$ detectors to detect $\mathcal{O}(N^2K)$ relationships.

Another key observation is that relationships are semantically related to each other. For example, a "person riding a horse" and a "person riding an elephant" are semantically similar because both `elephant` and `horse` are animals. Even if we haven't seen many examples of "person riding an elephant", we might be able to infer it from "person riding a horse". Word vectors [158] naturally lend themselves in linking such relationships because they capture semantic similarity in language (*e.g.*`elephant` and `horse` are cast close together in a word vector space). Therefore, we also propose a **knowledge transfer module** that uses pre-trained word vectors [158] to cast relationships into a vector space where similar relationships are optimized to be close to each other. Using this semantic space, we can transfer knowledge from frequent relationships and improve the prediction of relationships that occur infrequently.

In this chapter, we propose a model that can learn to detect visual relationships by learning (1) visual appearance models for its objects and predicates and (2) transferring knowledge using the

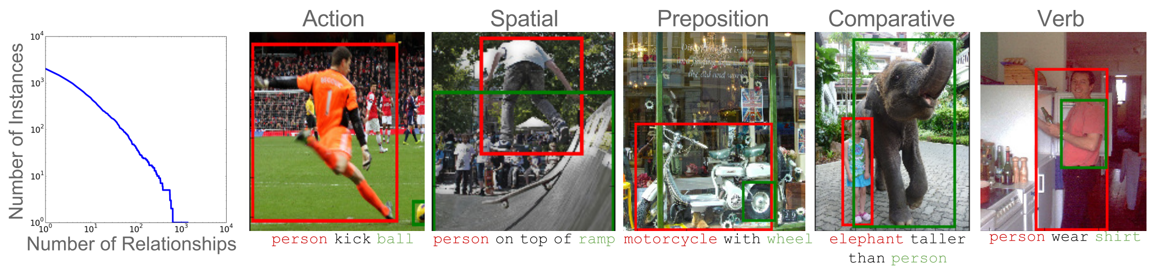


Figure 6.3: (left) A log scale distribution of the number of instances to the number of relationships in our dataset. Only a few relationships occur frequently and there is a long tail of infrequent relationships. (right) Relationships in our dataset can be divided into many categories, 5 of which are shown here: verb, spatial, preposition, comparative and action.

relationship vector space learnt from word vectors [158]. We train our model by optimizing a bi-convex function. To benchmark the task of visual relationship detection, we introduce a new dataset that contains 5000 images with 37,993 relationships. Existing datasets that contain relationships were designed for improving object detection [203] or image retrieval [100] and hence, don't contain sufficient variety of relationships or predicate diversity per object class. Our model outperforms all previous models in visual relationship detection. We further study how our model can be used to perform zero shot visual relationship prediction. Finally, we demonstrate that understanding relationships can improve image-based retrieval.

6.2 Related Work

Visual relationship prediction involves detecting the objects that occur in an image as well as understanding the interactions between them. There has been a series of work related to improving object detection by leveraging **object co-occurrence** statistics [155, 204, 120, 185, 65, 64]. Structured learning approaches have improved scene classification along with object detection using hierarchial contextual data from co-occurring objects [30, 94, 57, 218]. Unlike these methods, we study the *context* or *relationships* in which these objects co-occur.

Some previous works have attempted to learn **spatial relationships** between objects [74, 65] to improve multi-class segmentation [74]. They have specifically attempted to learn four spatial relationships: "above", "below", "inside", and "around" [65]. While we believe that that learning spatial relationships is important, we also study non-spatial relationships such as `pull` (actions), `taller than` (comparative), etc.

There have been numerous efforts in **human-object interaction** [192, 257, 144] and action recognition [79]. These efforts have attempted to learn discriminative models that distinguish between relationships where object$_1$ is a human, *e.g.*"playing violin" [256]. Visual relationship prediction is more general as $\texttt{object}_1$ is not constrained to be a human and the `predicate` doesn't

	Visual Phrases [203]	Scene Graph [100]	Ours
Images	2,769	5,000	5,000
Relationship types	13	23,190	6,672
Relationship instances	2,040	109,535	37,993
Predicates per object	120	2.3	24.25

Table 6.1: Comparison between our visual relationship benchmarking dataset with existing datasets that contain relationships.

have to be a verb.

Visual relationships are not a new concept. Some papers explicitly collected relationships in images [187, 76, 189, 233, 260] and videos [189, 117, 273] and helped models map these relationships from images to language. Relationships have also improved object localization [78, 118, 203, 200]. A meaning space of relationships have aided the cognitive task of mapping images to captions [49, 6, 86, 47]. Finally, relationships have been used to generate indoor images from sentences [24] and also to improve image search [100, 209]. In this chapter, we formalize visual relationship prediction as a task onto itself and demonstrate further improvements in image retrieval.

The most recent attempt at relationship prediction has been in the form of visual relationships called **visual phrases**. Learning appearance models for visual phrases have shown to improve individual object detection, i.e. detecting "a person riding a horse" improves the detection and localization of "person" and "horse" [203, 33]. Unlike our model, all previous works have attempted to detect only a handful of visual relationships and do not scale because most relationships are infrequent. We propose a model that manages to scale and detect millions of types of relationships. Additionally, our model is able to detect unseen relationships.

6.3 Visual Relationship Dataset

Visual relationships put objects in context; they capture the different interactions between pairs of objects. These interactions (shown in Figure 6.3) might be verbs (*e.g.*`wear`), spatial (*e.g.*`on top of`), prepositions (*e.g.*`with`), comparative (*e.g.*`taller than`), actions (*e.g.*`kick`) or a preposition phrase (*e.g.*`drive on`). A dataset for visual relationship prediction is fundamentally different from a dataset for objects detection. A relationship dataset should contain more than just objects localized in images; it should capture the rich variety of interactions between pairs of objects (predicates per object category). For example, a `person` can be associated with predicates such as `ride`, `wear`, `kick` etc. Additionally, the dataset should contain a large number of possible relationships (relationship types).

Existing datasets that contain relationships were designed to improve object detection [203] or image retrieval [100]. The Visual Phrases [203] dataset focuses on 13 relationship types. But, our

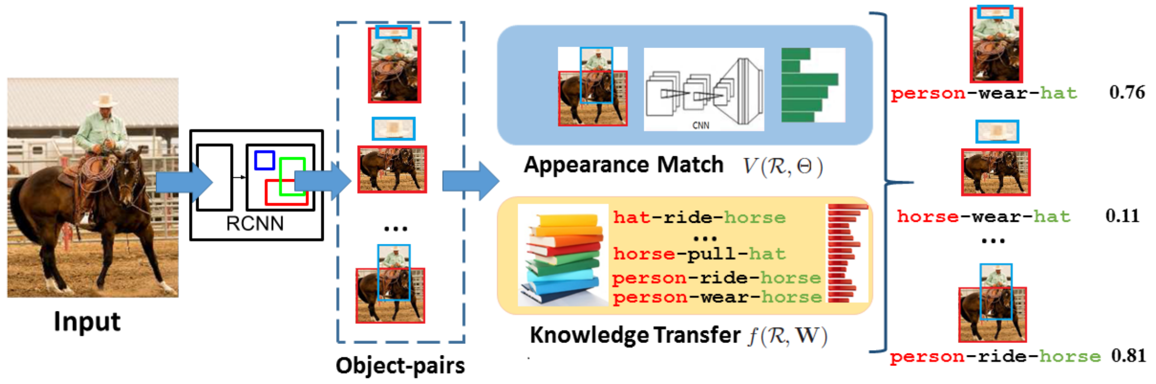


Figure 6.4: A overview of our visual relationship detection pipeline. Given an image as input, RCNN [72] generates a set of object proposals. Each pair of object proposals is then scored using a (1) visual appearance module and a (2) knowledge transfer module [158]. These scores are then thresholded to output a set of relationship labels (*e.g.riding*(*person*, *horse*)). Both objects in a relationship (*e.g.*`person` and `horse`) are localized as bounding boxes. The parameters of those two modules (W and Θ) are iteratively learnt in Chapter 6.4.1.

goal is to understand the rich variety of infrequent relationships. On the other hand, even though the Scene Graph dataset [100] has 23,190 relationship types [2], it only has 2.3 predicates per object. Detecting relationships on the Scene Graph dataset [100] essentially boils down to object detection. Therefore, we designed a dataset specifically for benchmarking visual relationship prediction.

Our dataset (Table 6.1) contains 5000 images with 100 object categories and 70 predicates. In total, the dataset contains 37,993 relationships with 6,672 relationship types and 24.25 predicates per object. Some example relationships are shown in Figure 6.3. The distribution of relationships in our dataset highlights the long tail of infrequent relationships (Figure 6.3(left)). We use 4000 images in our training set and test on the remaining 1000 images. 1,877 relationships occur in the test set but never occur in the training set.

6.4 Visual Relationship Prediction Model

The goal of our model is to detect visual relationships from an image. During training (Chapter 6.4.1), the input to our model is a fully supervised set of images with relationship annotations where the objects are localized as bounding boxes and labelled as *predicate*(*object*$_1$, *object*$_2$). At test time (Chapter 6.4.2), our input is an image with no annotations. We predict multiple relationships and localize the objects as bounding boxes in the image. Figure 6.4 illustrates a high level overview of our detection pipeline.

[2]Note that the Scene Graph dataset [100] was collected using unconstrained language, resulting in multiple annotations for the same relationship (*e.g.kick*(*man*, *ball*) and *is kicking*(*person*, *soccer ball*)). Therefore, 23,190 is an inaccurate estimate of the number of unique relationship types in their dataset.

6.4.1 Training Approach

In this section, we describe how we train our visual appearance module (Chapter 6.4.1) and knowledge transfer module (Chapter 6.4.1). Both the modules are combined together in our objective function (Chapter 6.4.1).

Visual Appearance Module

While Visual Phrases [203] learned a separate detector for every single relationship, we model the appearance of visual relationships $V()$ by learning the individual appearances of its comprising objects and predicate. While relationships are infrequent in real world images, the objects and predicates can be learnt as they individually occur more frequently. Furthermore, we demonstrate that our model outperforms Visual Phrases [203] in Table 6.2.

First, we finetune a convolutional neural network (CNN) (VGG net [217]) to classify each of our $N = 100$ objects. Similarly, we train a second CNN (VGG net [217]) to classify each of our $K = 70$ predicates using the union of the bounding boxes of the two participating objects in that relationship. Now, for each ground truth relationship $R_{\langle i,k,j\rangle}$ where i and j are the object classes (with bounding boxes O_1 and O_2) and k is the predicate class, we model V (shown in Figure 6.4) as:

$$\begin{aligned} &V(R_{\langle i,k,j\rangle}, \Theta | \langle O_1, O_2\rangle) = \\ &\qquad P_i(O_1)(\mathbf{z}_k^T \text{CNN}(O_1, O_2) + s_k) P_j(O_2) + \kappa_k G_k(O_1, O_2) \end{aligned} \tag{6.1}$$

where Θ is the parameter set of $\{\mathbf{z}_k, s_k, \kappa_k\}, k = 1, \ldots, K$. $P_i(O_1)$ and $P_j(O_2)$ are the CNN likelihoods of categorizing box O_1 as i and box O_2 as j. $\text{CNN}(O_1, O_2)$ is the predicate CNN features extracted from the union of the O_1 and O_2 boxes.

To model the spatial relationship $(G_k(O_1, O_2))$ between the boxes O_1 and O_2, we train a Gaussian Mixture Model (GMM) learnt using a 4 *dim.* feature representation: $\{\frac{x_{o1}-x_{o2}}{L}, \frac{y_{o1}-y_{o2}}{H}, \frac{A_{o1}}{A_{o1\cup o2}}, \frac{A_{o2}}{A_{o1\cup o2}}\}$, where A_{o1}, A_{o2} and $A_{o1\cup o2}$ are the area of O_1, O_2 and $O_1 \cup O_2$, respectively; H and L are the height and width of $O_1 \cup O_2$; x_{o1}, y_{o1} and x_{o2}, y_{o2} are the centroid of the O_1 and O_2 respectively.

Knowledge Transfer Module

One of our key observations is that relationships are semantically related to one another. For example, *ride*(*person*, *horse*) is semantically similar to *ride*(*person*, *elephant*). Even if we have not seen any examples of *ride*(*person*, *elephant*), we should be able to infer it from similar relationships that occur more frequently (*e.g.ride*(*person*, *horse*)). Our knowledge transfer module projects relationships into a relationship vector space where similar relationships are optimized to be close together. We first describe the function that projects a relationship to the vector space and then explain how we train this function.

Relationship projection function First, we use pre-trained word vectors [158] to cast the two objects in a relationship into an word embedding space [158]. Next, we concatenate these two vectors together and transform it into the relationship vector space using a learnt projection parameterized by $\mathbf{W}$. These projections present how two objects interact with each other under different predicates. We denote *w2vec*() as the function that converts a word to its 300 *dim.* vector. The relationship projection function (shown in Figure 6.4) is defined as:

$$f(\mathcal{R}_{\langle i,k,j\rangle}, \mathbf{W}) = \mathbf{w}_k^T[w2vec(t_i), w2vec(t_j)] + b_k \tag{6.2}$$

where t_j is the word (in text) of the j^{th} object category. $\mathbf{w}_k$ is a 600 *dim.* vector and b_k is a bias term. $\mathbf{W}$ is the set of $\{\{\mathbf{w}_1, b_1\}, \dots, \{\mathbf{w}_k, b_k\}\}$, $f(\langle i,k,j\rangle, \{\mathbf{w}_k, b_k\})$ where each row presents one of our K predicates.

Knowledge Transfer We want to optimize the projection function $f()$ such that it projects similar relationships closer to one another. We formulate this by using a heuristic where the distance between two relationships is proportional to the distance between its component objects and predicate:

$$[f(\mathcal{R}, \mathbf{W}) - f(\mathcal{R}', \mathbf{W})]^2 \propto d(\mathcal{R}, \mathcal{R}'), \quad \forall \mathcal{R}, \mathcal{R}' \tag{6.3}$$

where $d(\mathcal{R}, \mathcal{R}')$ is the sum of the cosine distances between of the two objects and the predicates of the two relationships $\mathcal{R}$ and $\mathcal{R}'$ in the pretrained word vector space [158]. We can rewrite Eq 6.3 in the form of a ratio as:

$$\rho_{\langle \mathcal{R}, \mathcal{R}'\rangle} = \frac{[f(\mathcal{R}, \mathbf{W}) - f(\mathcal{R}', \mathbf{W})]^2}{d(\mathcal{R}, \mathcal{R}')} \tag{6.4}$$

Now, to satisfy Eq 6.3, we expect the ratio in Eq 6.4 to be constant for all pairs of relationships. So, we randomly sample pairs of distinct relationships ($\langle \mathcal{R}, \mathcal{R}'\rangle$) and minimize their variance:

$$K(\mathbf{W}) = var(\{\rho_{\langle \mathcal{R}, \mathcal{R}'\rangle}(\mathbf{W}) \ \ \forall \mathcal{R}, \mathcal{R}'\}) \tag{6.5}$$

where $var()$ is a variance function.

Likelihood of a Relationship The output of our projection function should also indicate the likelihood of a visual relationship. For example, our model should not assign a high likelihood score to a relationships like *drive*(*dog*, *car*). We model this by enforcing that if $\mathcal{R}$ occurs more frequently than $\mathcal{R}'$ in our training data, then it should have a higher likelihood of occurring again. We formulate this as a rank loss function:

$$L(\mathbf{W}) = \sum_{\{\mathcal{R}, \mathcal{R}'\}} \max\{f(\mathcal{R}, \mathbf{W}) - f(\mathcal{R}', \mathbf{W}) + 1\} \tag{6.6}$$

While we only enforce this likelihood prior for the relationships that occur in our training data, the project function $f()$ generalizes it for all *predicate*(*object*$_1$, *object*$_2$) combinations, even if they are not present in our training data.

Objective function

So far we have presented our visual appearance module ($V()$) and the knowledge transfer module ($f()$) indicating likelihood of relationship. They are combined to maximize the rank of the ground truth relationship $\mathcal{R}$ with bounding boxes O_1 and O_2 using the following rank loss function:

$$\begin{aligned} C(\Theta, \mathbf{W}) = & \sum_{\langle O_1 O_2 \rangle, \mathcal{R}} \max\{1 + V(\mathcal{R}, \Theta | \langle O_1, O_2 \rangle) f(\mathcal{R}, \mathbf{W}) \\ & - \max_{\langle O_1', O_2' \rangle \neq \langle O_1, O_2 \rangle, \mathcal{R}' \neq \mathcal{R}} V(\mathcal{R}', \Theta | \langle O_1', O_2' \rangle) f(\mathcal{R}', \mathbf{W}), 0\} \end{aligned} \tag{6.7}$$

Therefore, our objective function combines Eq 6.7 with Eqs 6.5 and 6.6:

$$\min_{\Theta, \mathbf{W}} \{C(\Theta, \mathbf{W}) + \lambda_1 L(\mathbf{W}) + \lambda_2 K(\mathbf{W})\}, \tag{6.8}$$

where λ_1 and λ_2 are hyper-parameters that were obtained though grid search to maximize performance on the validation set. Note that both Eqs 6.7 and 6.6 are convex functions. Eq 6.5 is a biqudratic function with respect to $\mathbf{W}$. So our objective function Eq 6.8 has a quadratic closed form. We perform SGD iteratively on Eqs 6.7 and 6.6. It usually converges in $20 \sim 25$ iterations.

6.4.2 Testing

At test time, we use RCNN to produce a set of candidate object proposals for every test image. Next, we use the parameters learnt from the visual appearance model (Θ) and the knowledge transfer module ($\mathbf{W}$) to predict visual relationships ($\mathcal{R}^*_{\langle i,k,j \rangle}$) for every pair of RCNN object proposals $\langle O_1, O_2 \rangle$ using:

$$\mathcal{R}^* = \arg\max_{\mathcal{R}} V(\mathcal{R}, \Theta | \langle O_1, O_2 \rangle) f(\mathcal{R}, \mathbf{W}) \tag{6.9}$$

6.5 Experiments

We evaluate our model by detecting visual relationships from images. We show that our proposed method outperforms previous state-of-the-art methods. We also measure how our model performs in zero-shot learning of visual relationships. Finally, we demonstrate that understanding visual relationship can improve common computer vision tasks like content based image retrieval.

6.5.1 Visual Relationship Prediction

Setup. Given an input image, our task is to extract a set of visual relationships $predicate(object_1, object_2)$ and localize the objects as bounding boxes in the image. We train our model using the 4000

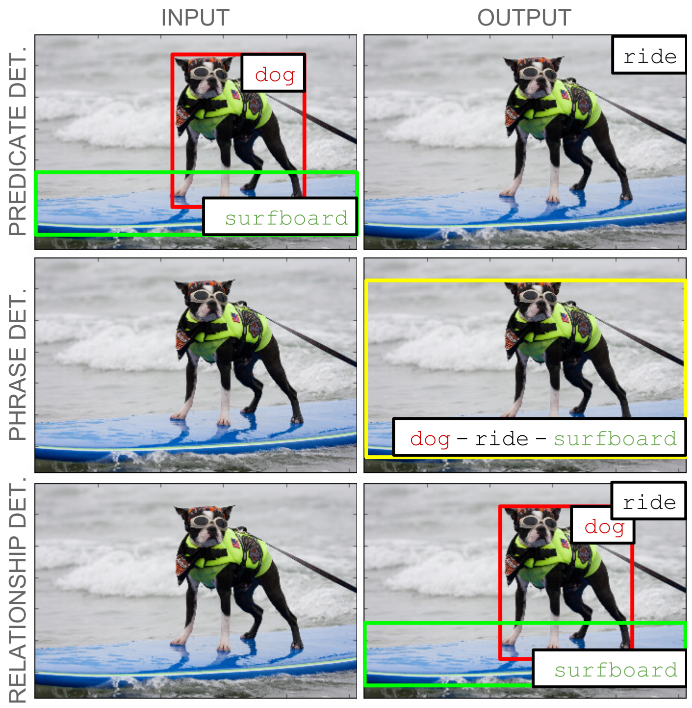


Figure 6.5: We evaluate visual relationship detection using three conditions: predicate detection (where we only predict the predicate given the object classes and boxes), phrase detection (where we label a region of an image with a relationship) and relationship detection (where we detect the objects and label the predicate between them).

	Visual Relationship Prediction (Chapter 6.5.1)						Zero Shot Prediction (Chapter 6.5.2)					
	Phrase Det.		Rel. Det.		Pred. Det.		Phrase Det.		Rel. Det.		Pred. Det.	
	R@100	R@50	R@100	R@50	R@100	R@50	R@100	R@50	R@100	R@50	R@100	R@50
Visual Phrases [203]	0.07	0.04	-	-	1.02	0.97	-	-	-	-	-	-
Joint CNN [217]	0.09	0.07	0.09	0.07	1.16	1.04	-	-	-	-	-	-
Ours - V only	3.92	3.47	4.02	3.87	8.45	7.03	0.85	0.73	0.72	0.63	1.49	1.05
Ours L only	0.08	0.08	0.08	0.08	17.43	14.21	0.01	0.00	0.01	0.00	1.23	1.17
Ours - V + naive FC	5.01	4.46	4.82	4.15	27.56	22.41	-	-	-	-	-	-
Ours - V + L only	6.87	6.44	7.86	7.05	34.17	29.69	2.76	2.54	2.45	2.13	4.57	4.02
Ours - V + L + K	13.65	11.17	12.46	10.68	44.28	41.26	4.71	4.42	4.65	4.21	8.21	7.84

Table 6.2: Results for visual relationship extraction. R@100 and R@50 are abbreviation Recall @ 100 and 50.

training images and perform visual relationship prediction on the 1000 test images.

The evaluation metrics we report is **recall @ 100** and **recall @ 50** [1]. We do not use mean average precision (mAP) scores because it is difficult to collect an exhaustive set of *all* possible relationships between *all* objects in an image. Consider the case where our model predicts *taller* *than*(*person*, *person*). Even if the prediction is correct, mAP would penalize the prediction if we do not have that ground truth annotation.

Detecting a visual relationship involves classifying both the objects, predicting the predicate and localization both the objects. To study how our model performs on each of these tasks, we measure visual relationship prediction under the following conditions:

1. In **predicate detection** (Figure 6.5(top)), our input is an image and set of localized objects. The task is to predict a set of possible predicates between pairs of objects. This condition allows us to study how difficult it is to predict relationships without the limitations of object detection [72].

2. In **phrase detection** (Figure 6.5(middle)), our input is an image and our task is to output a label $predicate(object_1,\ object_1)$ and localize the entire relationship as *one* bounding box having at least 0.5 overlap with ground truth box. This is the evaluation used in Visual Phrases [203].

3. In **relationship detection** (Figure 6.5(bottom)), our input is an image and our task is to output a set of $predicate(object_1,\ object_1)$ and localize *both* object$_1$ and object$_2$ in the image having at least 0.5 overlap with their ground truth boxes simultaneously.

Comparison Models We compare our method with some state-of-that-art approaches [203, 217]. We further perform ablation studies on our model, considering just the visual appearance, the likelihood term and knowledge transfer components to study their contributions.

- **Visual Phrases.** Similar to Visual Phrases [203], we train deformable parts models for each of the 6,672 relationships (*e.g.*"chair under table") in our training set.
- **Joint CNN.** We train a CNN model [217] to predict the three components of a relationship together. Specifically, we train a 270 (100+100+70) way classification model that learns to score the two objects (100 categories each) and predicate (70 categories).
- **Visual Appearance (Ours - V only).** We only use the visual appearance module of our model described in Eq 6.7 by optimizing $V()$.
- **Likelihood of a Relationship (Ours - L only).** We only use the likelihood of a relationship described in Eq 6.6 by optimizing $L()$.
- **Visual Appearance + Naive Frequency (Ours - V + naive FC).** One of the contributions of our model is the ability to transfer knowledge via our semantic projection function $f()$ (Eq 6.2). Here, we replace $f()$ with a function that maps a relationship to its frequency in our training data. Using this naive function, we hope to test the effectiveness of $f()$.
- **Visual Appearance + Likelihood (Our - V + L only).** We use both the visual appearance module (Eq 6.7) and the likelihood term (Eq 6.6) by optimizing both $V()$ and $L()$. The only part of our model missing is $K()$ Eq 6.5, which projects similar relationships closer.

- **Full Model (Ours - V + L + K).** This is our full model. It contains the visual appearance module (Eq 6.7), the likelihood term (Eq 6.6) and the knowledge transferred (Eq 6.5) from similar relationships.

Results. In Table 6.2, Visual Phrases [203] and Joint CNN [217] train an individual detector for every relationship. Since the space of all possible relationships is large (we have 6,672 relationship types in the training set), there is a shortage of training examples for infrequent relationships, causing both models to perform poorly on predicate, phrase and relationship detection. (Ours - V only) can't can't discriminative between similar relationships by itself resulting 4.2 R@100 for relationship detection. Similarly, (Ours - L only) always predicts the most frequent relationship *wear*(*person*, *shirt*) and results in recall of 0.08 R@100 which is the percentage of that relationship in our testing data. These problems are remedied when both V and L are combined in (Ours - V + L only) with an increase of 3% R@100 in on both phrase and relationship detection and more than 10% increase in predicate detection. We notice that (Ours - V + Naive FC) performs worse than (Ours - V + L only) and (Ours - V + L + K). (V + Naive FC) is missing our relationship projection function $f()$, which learns to predict the likelihood of a predicted relationship. Also, we observe that (Ours - V + L + K) has an 11% improvement in comparison to (Ours - V + L only) in predicate detection, demonstrating that knowledge transfer from similar relationships significantly helps improve visual relationship detection. By comparing the performance of all the models between relationship and predicate detection, We notice a 30% R@100 drop. This drop in recall is largely because we have to localize two objects simultaneously, amplifying the object detection errors. Note that even when we have ground truth object proposals (in predicate detection), predicting relationships is still a difficult task with 44.28 at R@100.

Qualitative Results. In Figure 6.6(a)(b)(c), Visual Phrase and Joint CNN incorrectly predict a common relationship: *drive*(*person*, *car*) and *next to*(*car*, *tree*). These models tend to predict the most common relationship as they see a lot of them during training. In comparison, our model correctly predicts and localizes the objects in the image. Figure 6.6(d)(e)(f) compares the various components of our model. Without the relationship likelihood score, (Ours - V only) incorrectly classifies a wheel as a clock in (d) and mislabels the predicate in (e) and (f). Without any visual priors, (Ours - L only) always reports the most frequent relationship *wear*(*person*, *shirt*). (Ours - V + L) fixes (d) by correcting the visual model's misclassification of the `wheel` as a `clock`. But it still does not predict the correct predicate for (e) and (f) because *ride*(*person*, *elephant*) and *hold*(*hand*, *phone*) rarely occur in our training set. However, our full model (Ours - V + L + K) leverages similar relationships it has seen before (e.g. *ride*(*man*, *horse*)) and is able to correctly detect the relationships in (e) and (f).

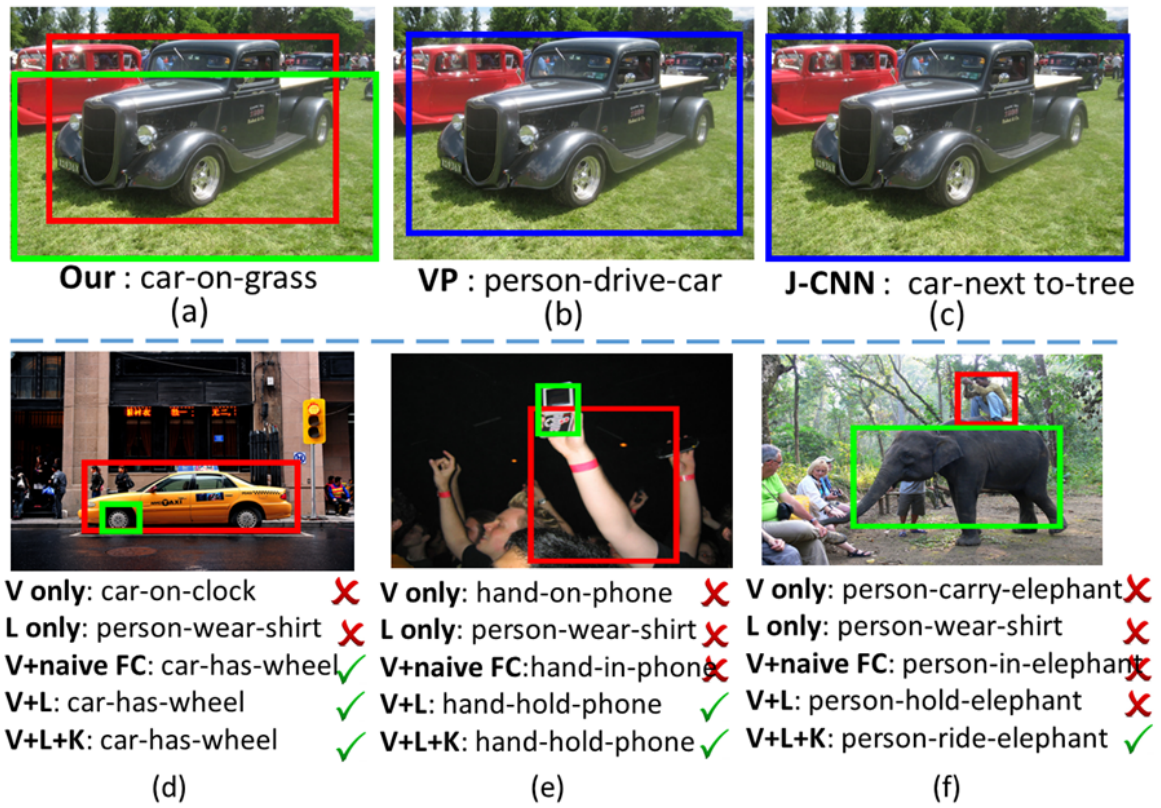

Our : car-on-grass
(a)
VP : person-drive-car
(b)
J-CNN : car-next to-tree
(c)
V only: car-on-clock ✗
L only: person-wear-shirt ✗
V+naive FC: car-has-wheel ✓
V+L: car-has-wheel ✓
V+L+K: car-has-wheel ✓
(d)
V only: hand-on-phone ✗
L only: person-wear-shirt ✗
V+naive FC:hand-in-phone ✗
V+L: hand-hold-phone ✓
V+L+K: hand-hold-phone ✓
(e)
V only: person-carry-elephant ✗
L only: person-wear-shirt ✗
V+naive FC: person-in-elephant ✗
V+L: person-hold-elephant ✗
V+L+K: person-ride-elephant ✓
(f)

Figure 6.6: Examples of comparative results. (a), (b) and (c) show our, Visual Phrases [203] and Joint CNN [217] results respectively on the same image. All ablation studies results for (d), (e) and (f) are report below corresponding image. Tick and cross mark the correct and incorrect result respectively. Phrase, object$_1$ and object$_2$ boxes are in blue, red and green respectively.

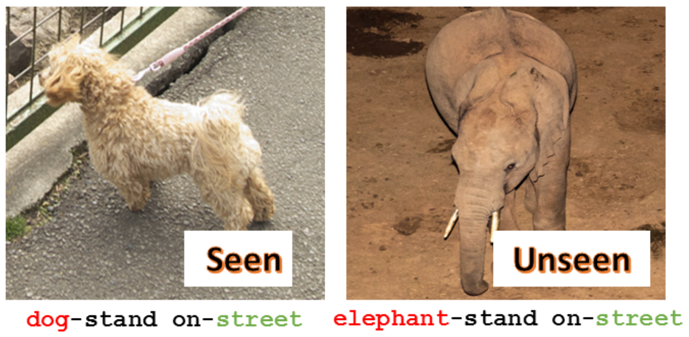


Figure 6.7: We can predict “elephant stand on street” (which we haven't seen in the training set) using similar seen relationships like “dog stand on street”.

6.5.2 Zero-shot Learning

Owing to the long tail of relationships in real world images, it is difficult to build a dataset with every possible relationship. Therefore, a model that detects visual relationships should also be able

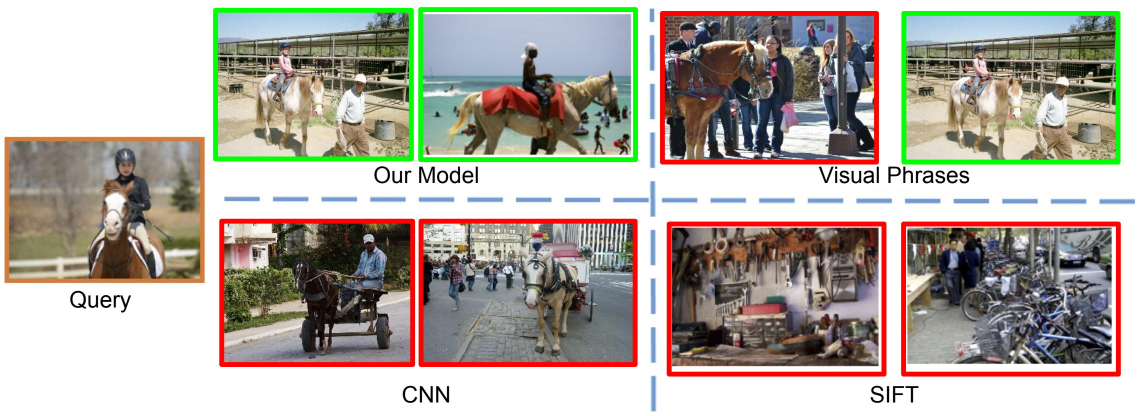


Figure 6.8: Examples retrieval results using an image as the query.

to perform zero-shot learning and predict relationships it has never seen before. Our model is able to leverage similar relationships it has already seen to detect unseen relationships.

Setup. Our test set contains 1,877 relationships that never occur in our training set (*e.g.stand on*(*elephant*, *street*)). These unseen relationships can be inferred by our model using similar relationships (*e.g.stand on*(*dog*, *street*)) from our training set (Figure 6.7). We report recall @ 50 and recall @ 100 for detecting unseen relationships in Table 6.2 (right) for predicate detection, phrase detection, and relationship detection.

Results. Since Visual Phrases [203], Joint CNN [217] and (Ours - V + Naive FC) require training examples to detect any unseen relationship, it is unable to perform zero shot learning. (Ours - V) achieves a low 1.49 R@100 because visual appearances are not discriminative enough to predict unseen relationships. (Ours - L only) performs poorly in predicate detection (1.23 R@100) because it automatically returns the most common predicate. Naturally, it performs worse for relationship detection (0.01 R@100). By comparing (Ours - V+ L+ K) and (Ours - V + L only), we find the use of K can gain an improvement of 30% since it is utilizing similar relationships to enable zero shot predictions.

6.5.3 Image based Retrieval

An important task in computer vision is image retrieval. An improved retrieval model should be able to infer the relationships between the objects in an image. We will demonstrate that the use of visual relationship prediction can improve the quality of the retrieved results.

Setup. Recall that our test set contains 1000 images. Every query uses 1 of these 1000 images and ranks the remaining 999. We use 54 query images in our experiments. Two annotators were asked to

	R@1	R@5	R@10	Median r
GIST [167]	0.00	5.60	8.70	68
SIFT [141]	0.70	6.10	10.3	54
CNN [217]	3.15	7.70	11.5	20
Visual Phrases [203]	8.72	18.12	28.04	12
Our Model	**10.82**	**30.02**	**47.00**	**4**

Table 6.3: Example image retrieval query using a image of a *ride*(*person*, *horse*). Note that a *higher* recall and *lower* median rank indicates better performance.

rank image results for each of the 54 queries. To avoid bias, we consider the results for a particular query as ground truth only if it was selected by both annotators. We evaluate our image retrieval performance using recall @ 1, recall @ 5 and recall @ 10 and median rank [100]. For comparison, we use three image descriptors that are commonly used in image retrieval: CNN [217], GIST [167] and SIFT [141]. We rank results for a query in ascending order of $L2$ distance from the query image.

Given a query image, our model predicts a set of visual relationships $\{R_1, \ldots, R_n\}$ with a probability of $\{P_1^q, \ldots, P_n^q\}$ respectively. Next, for every image I_i in our test set, it predicts $R_1, \ldots, R_n$ with a confidence of $\{P_1^i, \ldots, P_n^i\}$. We calculate a matching score between an image with the query as $\sum_{j=1}^{n} P_j^q * P_j^i$. We also compare our model with Visual Phrases' [203].

Results. SIFT [141] and GIST [167] descriptors perform poorly with a median rank of 54 and 68 (Table 6.3) because they measure structural similarity between images and not the semantic contents themselves. The CNN [217] descriptor can capture object-level information, so it performs much better with a median rank of 20. However, it still fails to understand visual relationships with higher-level semantic information. Our method captures the visual relationships present in the query image, which is important for high quality image retrieval, improving retrieval with a median rank of 4.

When queried using an image of a ”person riding a horse” (Figure 6.8), SIFT [141] returns images that are visually similar but are not semantically relevant. CNN [217] retrieves one image that contains a horse and one that contains both a man and a horse but neither of them capture the relationship: “person riding a horse”. Visual Phrases [203] and our model are able to detect the relationship *ride*(*person*, *horse*) and perform better.

6.6 Conclusion

In this chapter, we have proposed a model to detect multiple visual relationships in a single image Our model learns to detect thousands of relationships even when there are very few training examples by (1) learning the visual appearance of the component objects and predicates of the relationship and (2) by utilizing knowledge transferred from similar relationships. This model outperforms previous state of the art [203] on visual relationship detection. We have also demonstrated that our model can be used for zero shot learning of visual relationships. We introduce a new dataset of 5,000 images and 37,993 relationships that can be used for further benchmarking. Finally, by understanding visual relationships, our model improves content based image retrieval.

Chapter 7

Semantic Image Retrieval

7.1 Image Retrieval using Scene Graphs

7.1.1 Introduction

Retrieving images by describing their contents is one of the most exciting applications of computer vision. An ideal system would allow people to search for images by specifying not only objects ("man", "boat") but also structured *relationships* ("man on boat") and *attributes* ("boat is white") involving these objects. Unfortunately current systems fail for these types of queries because they do not utilize the structured nature of the query, as shown in Fig. 7.1.

To solve this problem, a computer vision system must explicitly represent and reason about the *objects*, *attributes*, and *relationships* in images, which we refer to as *detailed semantics*. Recently Zitnick et al. have made important steps toward this goal by studying *abstract scenes* composed of clip-art [272, 273, 63]. They show that perfect recognition of detailed semantics benefits image understanding and improves image retrieval.

Bringing this level of semantic reasoning to *real-world scenes* would be a major leap forward, but doing so involves two main challenges: *(1)* interactions between objects in a scene can be highly complex, going beyond simple pairwise relations, and *(2)* the assumption of a closed universe where all classes are known beforehand does not hold.

In order to address these challenges, this chapter proposes a novel framework for detailed semantic image retrieval, based on a conditional random field (CRF [122]) model of visual scenes. Our model draws inspiration from recent work in computer graphics that uses graph-based formulations to compare [59] and generate [23] scenes. We use the notion of a *scene graph* to represent the detailed semantics of a scene.

Our scene graphs capture the detailed semantics of visual scenes by explicitly modeling objects,

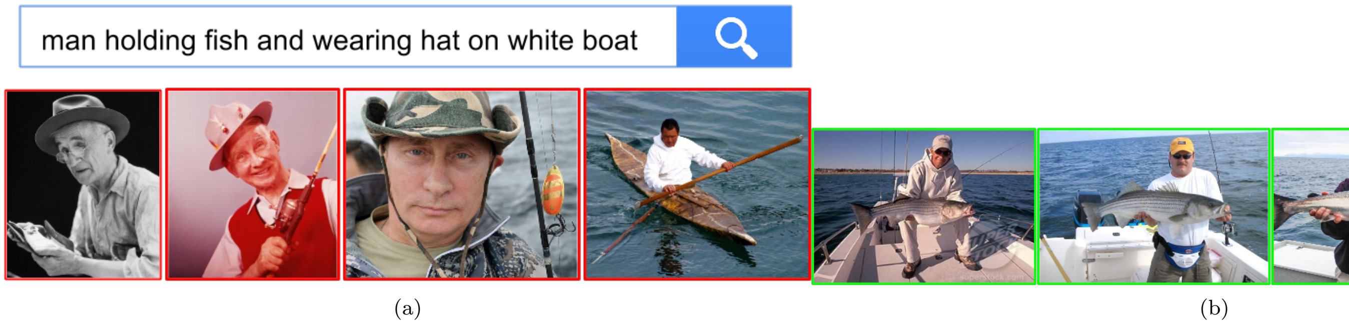


Figure 7.1: Image search using a complex query like "man holding fish and wearing hat on white boat" returns unsatisfactory results in (a). Ideal results (b) include correct *objects* ("man", "boat"), *attributes* ("boat is white") and *relationships* ("man on boat").

attributes of objects, and relationships between objects. Our model performs semantic image retrieval using scene graphs as queries. Replacing textual queries with scene graphs allows our queries to describe the semantics of the desired image in precise detail without relying on unstructured text. This formulation is related to a number of methods for object and scene recognition using context [78, 42]. But by using scene graphs, we can model multiple modes of interaction between pairs of objects while traditional CRF models are more restricted, and encode a fixed relation given two nodes (e.g. think of "dog [eating, or playing, or being on the right of] a keyboard" in a scene graph versus. dog *on the right of* a keyboard in a CRF).

Specifically, our contributions are:

- We introduce a CRF model (Chapter 7.1.5) for semantic image retrieval using scene graph (Chapter 7.1.3) queries. We show that our model outperforms baseline models that reason only about objects, and simple content-based image retrieval methods based on low-level visual features (Chapter 7.1.6). This addresses challenge (1) above.

- We introduce a novel dataset[1] (Chapter 7.1.4) of $5,000$ human-annotated scene graphs grounded to images that use an open-world vocabulary to describe images in great detail, addressing challenge (2) above.

We set out to demonstrate the importance and utility of modeling detailed semantics using a scene graph representation for an image retrieval task. But as our experiments demonstrate, an advantage of this representation is its ability to offer deeper and detailed insights into images in a general framework. Our model can perform semantically meaningful image retrieval based on an entire scene (Chapter 7.1.6), or only parts of scenes (Chapter 7.1.6), and localizes specific objects (Chapter 7.1.6). This differentiates our intention and system from traditional content-based image retrieval work, which is not the focus of our chapter.

[1]Available at the first author's website.

7.1.2 Related Works

Image retrieval. Content-based image retrieval methods typically use low-level visual feature representations [177, 21], indexing [35, 265, 96, 97, 228], efficient sub-image search [124], object-category based retrieval [8, 238, 7], and other methods of modeling image similarity [184] to retrieve images based on their contents.

Text-based scene representations. There has been much recent interest in models that can jointly reason about images and natural language descriptions, ranging from Flickr tags [135] over sentences with canonical structure [117, 161, 273] to unaligned text corpora [222, 107, 225, 262]. While these models achieve impressive results in scene classification and object recognition using only weak supervision, they are typically limited in terms of expressiveness.

In contrast, our scene graphs are a structured representation of visual scenes. Each node is explicitly grounded in an image region, avoiding the inherent referential uncertainty of text-based representations. We currently use strong supervision in the form of a crowd-sourced data set (Chapter 7.1.4), but ultimately envision a system that learns novel scene graphs via active learning, like NEIL [26] or LEVAN [43].

Structured scene representations. More structured representations of visual scenes explicitly encode certain types of properties, such as attributes [55, 50, 48, 172, 123], object co-occurrence [156], or spatial relationships between pairs of objects [31, 42, 203, 32]. Our scene graphs generalize these representations, since they allow us to express each of them in a unified way (Chapter 7.1.3). Concretely, our CRF (Chapter 7.1.5) learns models for particular relations between pairs of objects, similar in spirit to [78] or [258]. However, we consider a much larger, open vocabulary of objects and relationships.

Graph-structured representations have attained wide-spread use in computer graphics to efficiently represent compositional scenes. Fisher et al. [60] use graph kernels to compare 3D scenes, and Chang et al. [23] generate novel 3D scenes from natural language descriptions using a scene graph representation. Parse graphs obtained in scene parsing [266, 255] are typically the result of applying a grammar designed for a particular domain (such as indoor scenes [266]), in contrast to our generic scene graphs.

Recent work by Lin et al. [138] constructs semantic graphs from text queries using hand-defined rules to transform parse trees and uses these semantic graphs to retrieve videos in the context of autonomous driving. Their system is constrained to the six object classes from KITTI [67] while our system uses an open-world vocabulary; our scene graphs also tend to be more complex than their semantic graphs.

Zitnick et al. have studied detailed semantics using abstract scenes built from clip-art aligned to a corpus of sentences [272, 273, 63]. Our Chapter extends this work as follows: first, we explicitly address real-world scenes, replacing the idealistic setting of perfect object and attribute recognition [272] by uncertain detections. Second, our scene graphs go beyond pairwise relations: we model and discover meaningful higher-order interactions between multiple objects and attributes (these sub-graphs can be seen as a generalization of visual phrases [203, 32]). Third, our CRF for scene graph grounding (Chapter 7.1.5) can ground a query scene graph to an unannotated test image.

Real-world scene datasets. Datasets have been a driving force of computer vision algorithms, ranging from classification [253] to object recognition and segmentation [52, 40, 201, 46]. More recently, there has been a shift away from iconic images toward datasets depicting objects in the context of entire scenes, e.g., SUN09 [31], PASCAL-Context [162], and the COCO [139] datasets. Our novel dataset of real-world scene graphs advances in this direction by adding attributes and relationships. In contrast to previous work, the current version of our dataset with $5,000$ images focuses on depth rather than breadth (Chapter 7.1.4), resulting in a level of semantic detail that has not been achieved before. While this level of detail is similar in spirit to the Lotus Hill dataset [261], our dataset is based on an open universe assumption and is freely available.

7.1.3 Scene Graphs

To retrieve images containing particular semantic contents, we need a formalized way of representing the contents of a scene. This representation must be powerful enough to describe the rich variety of scenes that can exist, without being too cumbersome. To this end, we define two abstractions: a *scene graph*, which is a way of describing a scene, and a *scene graph grounding*, which is a concrete association of a scene graph to an image.

Definition

A *scene graph* is a data structure that describes the contents of a scene. A scene graph encodes object instances, attributes of objects, and relationships between objects.

This simple formulation is powerful enough to describe visual scenes in great detail because it places no restriction on the types of objects, attributes, and relationships that can be represented. Fig. 7.2 (bottom) shows an example of a scene graph. In this example we see that object instances may be people ("girl"), places ("tennis court"), things ("shirt"), or parts of other objects ("arm"). Attributes can describe color ("cone is orange"), shape ("logo is round"), and pose ("arm is bent"). Relationships can encode geometry ("fence behind girl"), actions ("girl swinging racket"), and object parts ("racket has handle").

Formally, given a set of object classes $\mathcal{C}$, a set of attribute types $\mathcal{A}$, and a set of relationship types $\mathcal{R}$, we define a scene graph G to be a tuple $G = (O, E)$ where $O = \{o_1, \ldots, o_n\}$ is a set of objects and $E \subseteq O \times \mathcal{R} \times O$ is a set of edges. Each object has the form $o_i = (c_i, A_i)$ where $c_i \in \mathcal{C}$ is the class of the object and $A_i \subseteq \mathcal{A}$ are the attributes of the object.

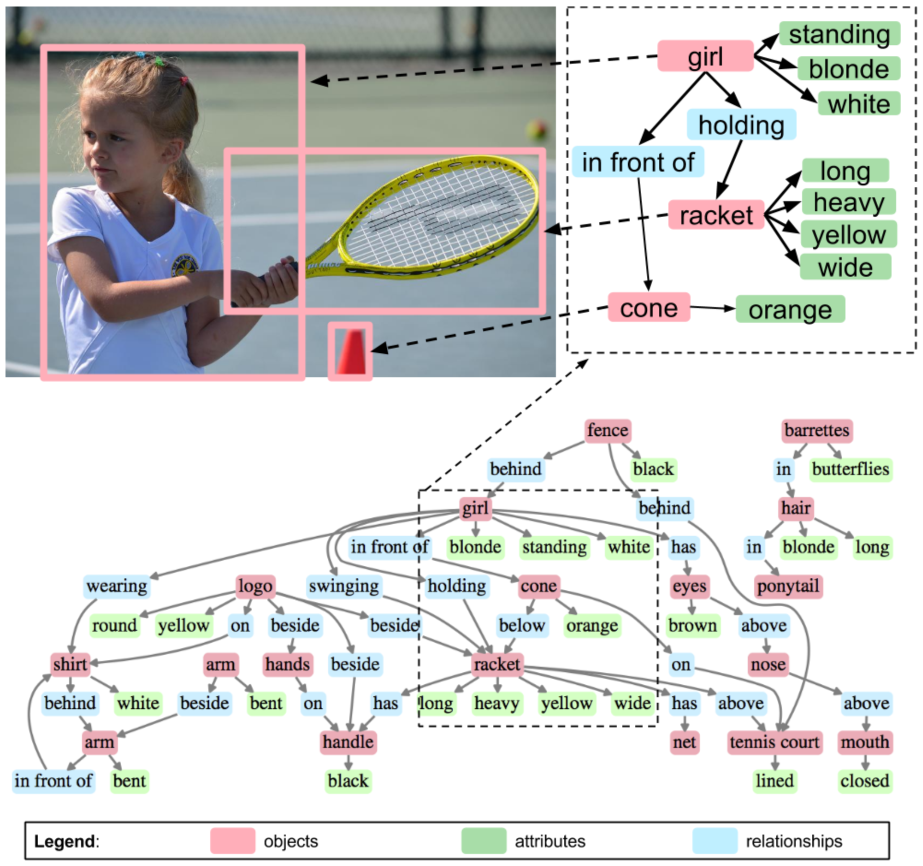


Figure 7.2: An example of a scene graph (bottom) and a grounding (top). The scene graph encodes objects ("girl"), attributes, ("girl is blonde"), and relationships ("girl holding racket"). The grounding associates each object of the scene graph to a region of an image. The image, scene graph, and grounding are drawn from our *real-world scene graphs* dataset (Chapter 7.1.4).

Grounding a scene graph in an image

A scene graph on its own is not associated to an image; it merely describes a scene that could be depicted by an image. However a scene graph can be *grounded* to an image by associating each object instance of the scene graph to a region in an image. Fig. 7.2 (top) shows an example of part of a scene graph grounded to an image.

Formally, we represent an image by a set of candidate bounding boxes B. A grounding of a scene graph $G = (O, E)$ is then a map $\gamma : O \to B$. For ease of notation, for $o \in O$ we frequently write $\gamma(o)$ as γ_o.

Given a scene graph and an image, there are many possible ways of grounding the scene graph to the image. In Chapter 7.1.5 we formulate a method for determining the best grounding of a scene graph to an image.

	Full dataset	Experiments Chapter 7.1.6	COCO 2014 [139]	ILSVRC 2014 (Det) [196]	Pascal VOC [46]
Object classes	**6,745**	**266**	80	200	20
Attribute types	3,743	145	-	-	-
Relationship types	1,310	68	-	-	-
Object instances	93,832	69,009	**886,284**	534,309	27,450
Attribute instances	110,021	94,511	-	-	-
Relationship instances	112,707	109,535	-	-	-
Instances per obj. class	13.9	259.4	**11,087.5**	2,672.5	1,372
Instances per attr. type	29.4	651.8	-	-	-
Instances per rel. type	86.0	1,610.8	-	-	-
Objects per image	**18.8**	**13.8**	7.2	1.1	2.4
Attributes per image	22.0	18.9	-	-	-
Relationships per image	22.5	21.9	-	-	-
Attributes per object	1.2	1.0	-	-	-
Relationships per object	2.4	2.3	-	-	-

Table 7.1: Aggregate statistics for our *real-world scene graphs* dataset, for the full dataset and the restricted sets of object, attribute, and relationship types used in experiments.

Why scene graphs?

An obvious alternative choice for representing the content of scenes is natural language. However, in order to represent visual scenes at the level of detail shown in Fig. 7.2, a full paragraph of description would be necessary:

A blonde white girl is standing in front of an orange cone on a lined tennis court and is holding a long heavy yellow wide racket that has a black handle. The girl is wearing a white shirt; there is a bent arm in front of the shirt and another bent arm beside the first. There is a round yellow logo on the shirt, and the logo is beside hands that are on the handle of the racket. There is a black fence behind the girl, and the girl has brown eyes above a closed mouth. There are butterflies barrettes in long blonde hair, and the hair is in a ponytail.

To make use of such a description for image retrieval, we would need to resolve co-references in the text [186, 102, 132], perform relationship extraction to convert the unstructured text into structured tuples [165], and ground the entities of the tuples into regions of the image described by the text [115]. Such pipelines are challenging even in constrained settings [115], and would not scale to text of the detail shown above.

We can avoid these complexities by working directly with grounded scene graphs. We find that with careful user interface design, non-expert workers can quickly construct grounded scene graphs of arbitrary complexity. Details can be found in Sec. 7.1.4 and in our supplementary material.

7.1.4 Real-World Scene Graphs Dataset

To use scene graphs as queries for image retrieval, we need many examples of scene graphs grounded to images. To our knowledge no such dataset exists. To this end, we introduce a novel dataset of

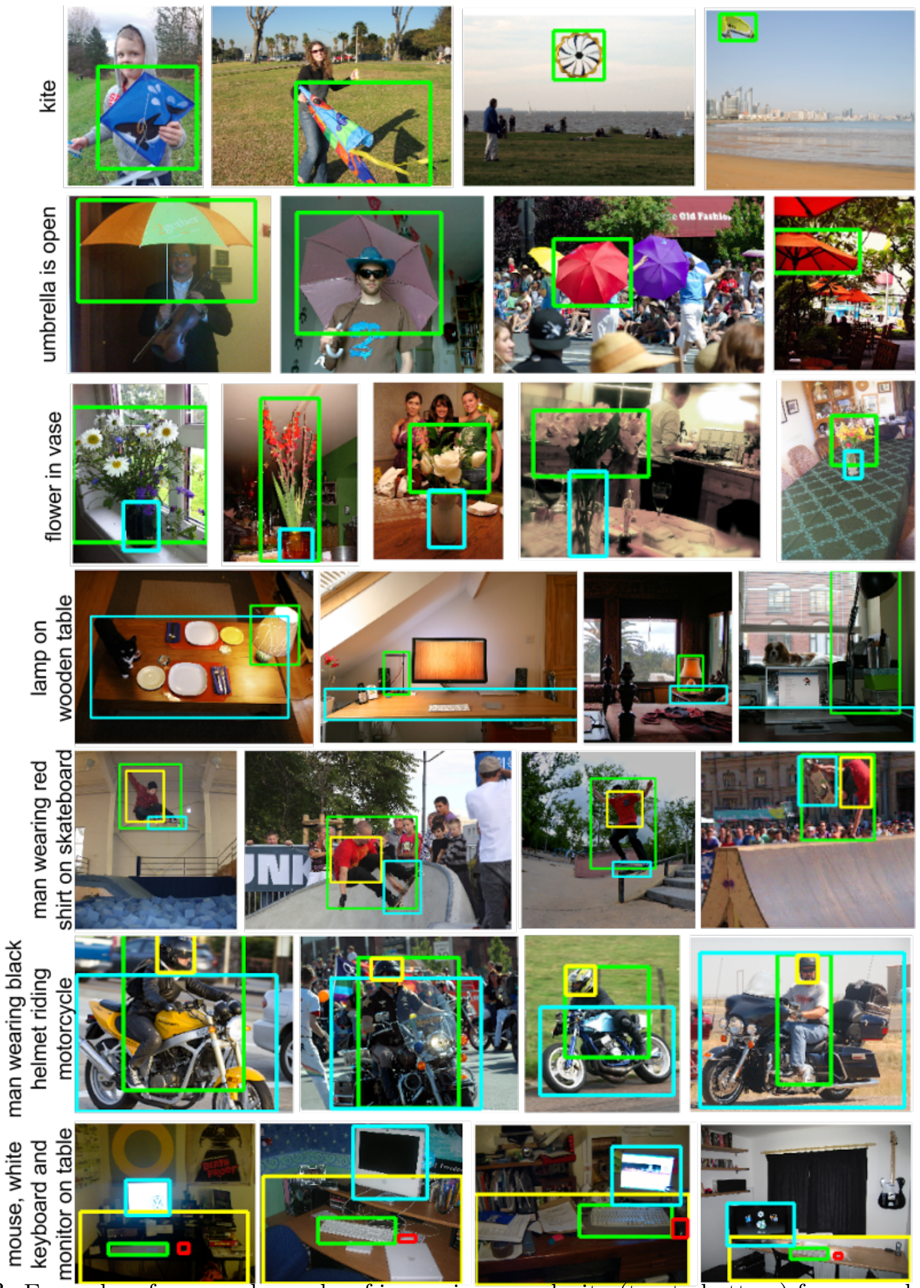

kite
umbrella is open
flower in vase
lamp on wooden table
man wearing red shirt on skateboard
man wearing black helmet riding motorcycle
mouse, white keyboard and monitor on table

Figure 7.3: Examples of scene sub-graphs of increasing complexity (top to bottom) from our dataset, with attributes and up to 4 different objects.

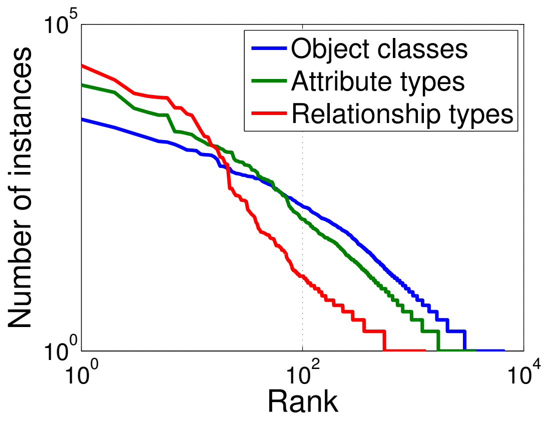


Figure 7.4: Objects, attributes and relations reveal a Zipf distribution when ordered by number of labeled instances.

real-world scene graphs.

Data collection

We manually selected 5,000 images from the intersection of the YFCC100m [235] and Microsoft COCO [139] datasets, allowing our dataset to build upon rather than compete with these existing datasets.

For each of these images, we use Amazon's Mechanical Turk (AMT) to produce a human-generated scene graph. For each image, three workers write (object, attribute) and (object, relationship, object) tuples using an open vocabulary to describe the image, and draw bounding boxes for all objects. Bounding boxes for objects are corrected and verified using a system similar to [230], and all tuples are verified by other workers. A detailed explanation of our data collection pipeline can be found in the supplementary material.

Analysis and statistics

Our full dataset of 5,000 images contains over 93,000 object instances, 110,000 instances of attributes, and 112,000 instances of relationships. For our experiments (Chapter 7.1.6), we consider only object classes and attribute types that appear at least 50 times in our training set and relationship types that occur at least 30 times in our training set. Even when we consider only the most common categories, as shown in Table 7.1, the dataset is still very rich with a mean of 13.8 objects, 18.9 attributes and 21.9 relationships per image.

A comparison of our dataset and other popular datasets can be found in Table 7.1. Compared to other datasets we prioritize detailed annotations of individual images over sheer quantity of annotated images. Our dataset contains significantly more labeled object instances per image than

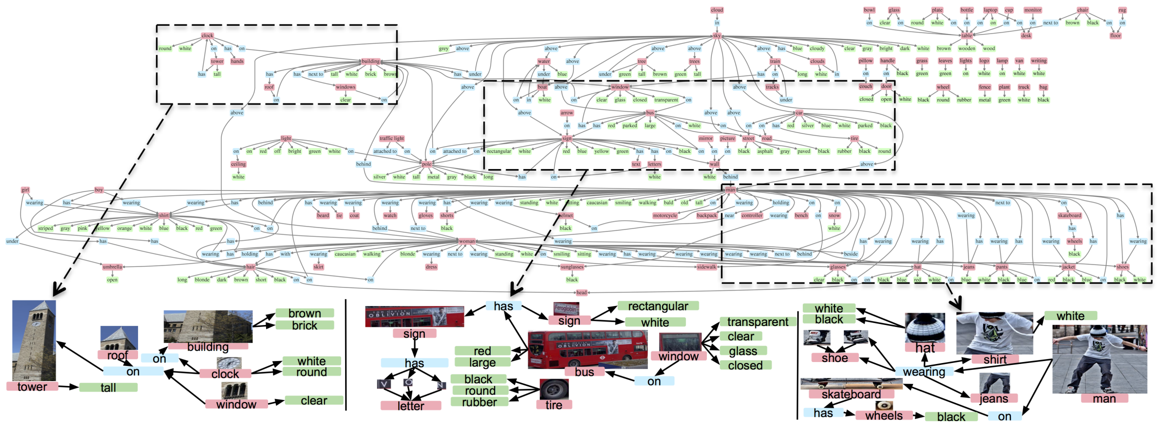


Figure 7.5: An *aggregate* scene graph computed using our entire dataset. We visualize the 150 most frequently occurring (object, relationship, object) and (object, attribute) tuples. We also provide 3 examples of scene graphs grounded in images that contribute to the sub-graphs within the dashed rectangles of the aggregated graph. Best viewed with magnification.

existing datasets, and also provides annotated attributes and relationships for individual object instances; these types of annotations are simply not available in other datasets. In addition, our decision to use an open vocabulary allows annotators to label the most meaningful features of each image instead of being constrained to a fixed set of predefined classes.

In contrast to previous work that looks at individual relations between pairs of objects in isolation [203], the deeply connected nature of our scene graph representation allows us to study object interactions of arbitrary complexity. Fig. 7.3 shows examples of scene-subgraphs that occur multiple times in our dataset, ranging from simple ("kite") to complex ("man on skateboard wearing red shirt"). These subgraphs also showcase the rich diversity expressed by our scene graphs. Attributes can for example encode object state ("umbrella is open"), material ("wooden table") and color ("red shirt", "black helmet"). Relationships can encode geometry ("lamp on table") as well as actions ("man riding motorcycle").

Fig. 7.5 uses our dataset to construct an *aggregate scene graph*, revealing the deeply connected nature of objects in the visual world. For example buses are frequently large and red, have black tires, are on streets, and have signs; signs can also appear on walls, which often occur behind men, who often wear white shirts and jeans. The high density of the aggregate scene graph around the "man", "woman", "sky", and "building" nodes suggest that these objects are prominent elements of the visual world, but for different reasons: sky and building occur in nearly every image, while the attributes and relationships of people in scenes carries a huge semantic weight.

7.1.5 Image Retrieval by Scene Graph Grounding

We wish to use a scene graph as a query to retrieve images portraying scenes similar to the one described by the graph. To do so, we need to measure the agreement between a query scene graph and an unannotated test image. We assume that this agreement can be determined by examining the best possible grounding of the scene graph to the image.

To this end we construct a conditional random field (CRF [122]) that models the distribution over all possible groundings. We perform maximum a posteriori (MAP) inference to find the most likely grounding; the likelihood of this MAP solution is taken as the score measuring the agreement between the scene graph and the image.

Reusing notation from Chapter 7.1.3, let $G = (O, E)$ be a scene graph, B be a set of bounding boxes in an image, and γ a grounding of the scene graph to the image. Each object $o \in O$ gives rise to a variable γ_o in the CRF, where the domain of γ_o is B and setting $\gamma_o = b \in B$ corresponds to grounding object o in the scene graph to box b in the image. We model the distribution over possible groundings as

$$P(\gamma \mid G, B) = \prod_{o \in O} P(\gamma_o \mid o) \prod_{(o,r,o') \in E} P(\gamma_o, \gamma_{o'} \mid o, r, o'). \tag{7.1}$$

We use Bayes' rule to rewrite the term $P(\gamma_o \mid o)$ as $P(o \mid \gamma_o)P(\gamma_o)/P(o)$. Assuming uniform priors over bounding boxes and object classes, $P(\gamma_o)$ and $P(o)$ are constants and can be ignored when performing MAP inference. Therefore our final objective has the form

$$\gamma^* = \arg\max_{\gamma} \prod_{o \in O} P(o \mid \gamma_o) \prod_{(o,r,o') \in E} P(\gamma_o, \gamma_{o'} \mid o, r, o'). \tag{7.2}$$

Unary potentials. The term $P(o \mid \gamma_o)$ in Equation 7.2 is a unary potential modeling how well the appearance of the box γ_o agrees with the known object class and attributes of the object o. If $o = (c, A)$ then we decompose this term as

$$P(o \mid \gamma_o) = P(c \mid \gamma_o) \prod_{a \in A} P(a \mid \gamma_o). \tag{7.3}$$

The terms $P(c \mid \gamma_o)$ and $P(a \mid \gamma_o)$ are simply the probabilities that the bounding box γ_o shows object class c and attribute a. To model these probabilities, we use R-CNN [72] to to train detectors for each of the $|\mathcal{C}| = 266$ and $|\mathcal{A}| = 145$ object classes and attribute types. We apply Platt scaling [179] to convert the SVM classification scores for each object class and attribute into probabilities.

Binary potentials. The term $P(\gamma_o, \gamma_{o'} \mid o, r, o')$ in Equation 7.2 is a binary potential that models how well the pair of bounding boxes $\gamma_o, \gamma_{o'}$ express the tuple (o, r, o'). Let $\gamma_o = (x, y, w, h)$ and

$\gamma_{o'} = (x', y', w', h')$ be the coordinates of the bounding boxes in the image. We extract features $f(\gamma_o, \gamma_{o'})$ encoding their relative position and scale:

$$f(\gamma_o, \gamma_{o'}) = \Big((x - x')/w,\ (y - y')/h,\ w'/w,\ h'/h\Big) \tag{7.4}$$

Suppose that the objects o and o' have classes c and c' respectively. Using the training data, we train a Gaussian mixture model (GMM) to model $P(f(\gamma_o, \gamma_{o'}) \mid c, r, c')$. If there are fewer than 30 instances of the tuple (c, r, c') in the training data then we instead fall back on an object agnostic model $P(f(\gamma_o, \gamma_{o'}) \mid r)$. In either case, we use Platt scaling to convert the value of the GMM density function evaluated at $f(\gamma_o, \gamma_{o'})$ to a probability $P(\gamma_o, \gamma_{o'} \mid o, r, o')$.

Implementation details

We compared the performance of several methods for generating candidate boxes for images, including Objectness [1], Selective Search (SS [239]), and Geodesic Object Proposals (GOP [112]). We found that SS achieves the highest object recall on our dataset; however we use GOP for all experiments as it provides the best trade-off between object recall ($\approx$ 70% vs $\approx$ 80% for SS) and number of regions per image (632 vs 1720 for SS). We perform approximate inference using off-the-shelf belief propagation [2].

7.1.6 Experiments

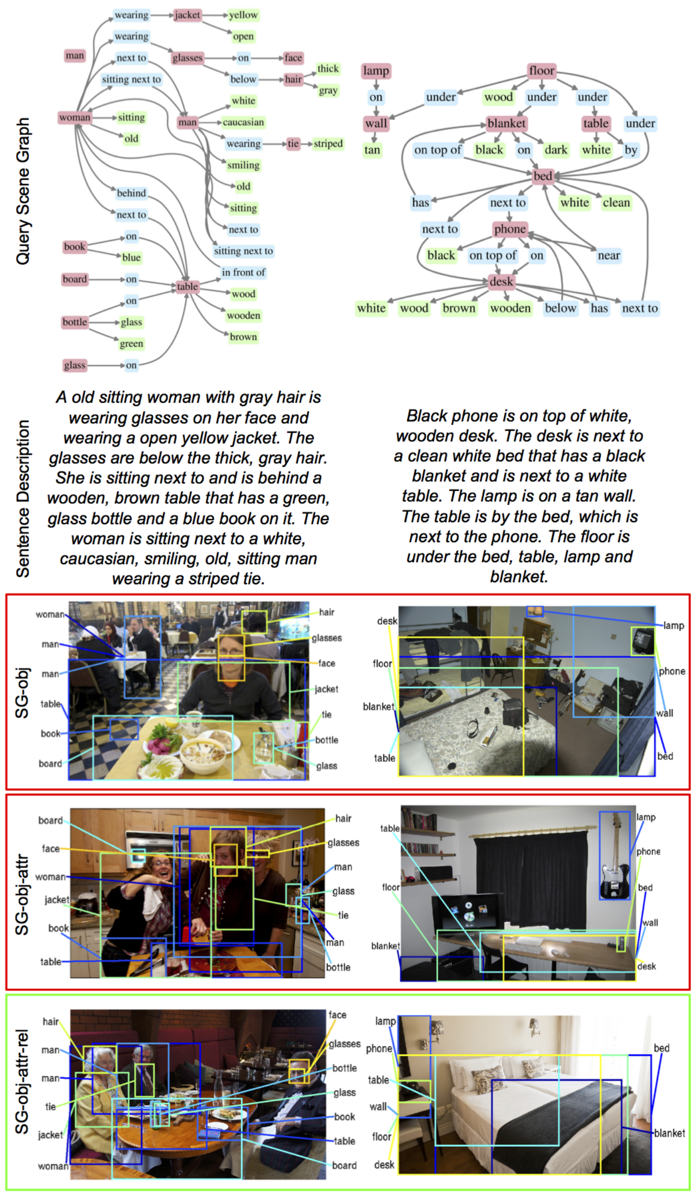
We perform image retrieval experiments using two types of scene graphs as queries. First, we use full ground-truth scene graphs as queries; this shows that our model can effectively make sense of extremely precise descriptions to retrieve images. Second, we jump to the other end of the query complexity spectrum and use extremely simple scene graphs as queries; this shows that our model is flexible enough to retrieve relevant images when presented with more open-ended and human-interpretable queries.

In addition, we directly evaluate the groundings found by our model and show that our model is able to take advantage of scene context to improve object localization.

Setup. We randomly split our dataset into 4,000 training images and 1,000 test images. Our final vocabulary for object classes and attribute types is selected by picking all terms mentioned at least 50 times in the training set, and our vocabulary for relationship types is the set of relationships appearing at least 30 times in the training set. Statistics of our dataset when restricted to this vocabulary are shown in the second column of Table. 7.1.

In our experiments we compare the following methods:

- **SG-obj-attr-rel** : Our model as described in Chapter 7.1.5. Includes unary object and attribute potentials and binary relationship potentials.



A old sitting woman with gray hair is wearing glasses on her face and wearing a open yellow jacket. The glasses are below the thick, gray hair. She is sitting next to and is behind a wooden, brown table that has a green, glass bottle and a blue book on it. The woman is sitting next to a white, caucasian, smiling, old, sitting man wearing a striped tie.

Black phone is on top of white, wooden desk. The desk is next to a clean white bed that has a black blanket and is next to a white table. The lamp is on a tan wall. The table is by the bed, which is next to the phone. The floor is under the bed, table, lamp and blanket.

Figure 7.6: Example results for retrieval using full scene-graph queries (Chapter 7.1.6). Top: Example query graphs. Middle: Rough textual equivalents of the query scene graphs. Bottom: Top-1 retrieval results with groundings for our 3 methods. In both cases SG-obj-attr-rel succeeds in ranking the correct image at rank 1.

- **SG-obj-attr** : Our model, using only object and attribute potentials.
- **SG-obj** : Our model, using only object potentials. Equivalent to R-CNN [72] since the object class potentials are rescaled R-CNN detection scores.

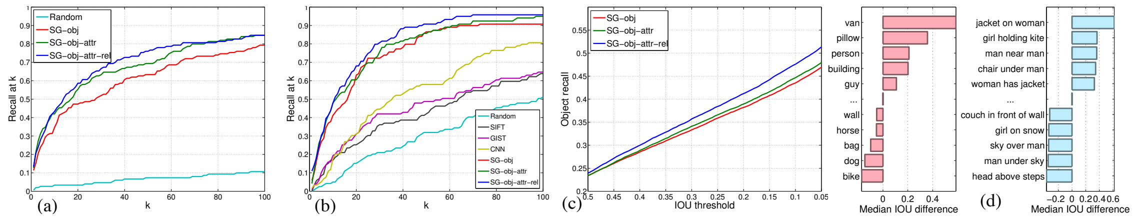


Figure 7.7: (a) Retrieval performance for entire scenes, (b) for partial scenes. (c) Object localization performance for entire scenes. (d) Increase in localization performance of our full model SG-obj-attr-rel vs SG-obj for individual objects (left) and objects participating in a relation (right). In (d), positive values indicate the SG-obj-attr-rel performs better than SG-obj .

- **CNN** [116]: L_2 distance between the last layer features extracted using the reference model from [98].
- **GIST** [167]: L_2 distance between the GIST descriptors of the query image and each test image (see Chapter 7.1.6).
- **SIFT** [141]: See Chapter 7.1.6.
- **Random**: A random permutation of the test images.

Full scene graph queries

		Rand	SIFT [141]	GIST [167]	CNN [116]	SG-obj [72]	SG-obj-attr	SG-obj-attr-rel
(a)	Med r	420	-	-	-	28	17.5	**14**
	R@1	0	-	-	-	0.113	0.127	**0.133**
	R@5	0.007	-	-	-	0.260	**0.340**	0.307
	R@10	0.027	-	-	-	0.347	0.420	**0.433**
(b)	Med r	94	64	57	36	17	12	**11**
	R@1	0	0	0.008	0.017	0.059	0.042	**0.109**
	R@5	0.034	0.084	0.101	0.050	0.269	0.294	**0.303**
	R@10	0.042	0.168	0.193	0.176	0.412	**0.479**	**0.479**
(c)	Med IoU	-	-	-	-	0.014	0.026	**0.067**
	R@0.1	-	-	-	-	0.435	0.447	**0.476**
	R@0.3	-	-	-	-	0.334	0.341	**0.357**
	R@0.5	-	-	-	-	0.234	0.234	**0.239**

Table 7.2: Quantitative results in entire scene retrieval ((a), Chapter 7.1.6), partial scene retrieval ((b), Chapter 7.1.6), and object localization ((c), Chapter 7.1.6).

We evaluate the performance of our model using the most complex scene graphs available to us – full human-generated scene graphs from our dataset. As argued in Chapter 7.1.3, these large scene graphs roughly correspond to a paragraph of text describing an image in great detail. Examples of query scene graphs and their rough textual equivalents are shown in the top half of Fig. 7.6.

Concretely, we select an image I_q and its associated human-annotated scene graph G_q from our test set. We use the graph G_q as a query to rank all of the test images, and record the rank of the

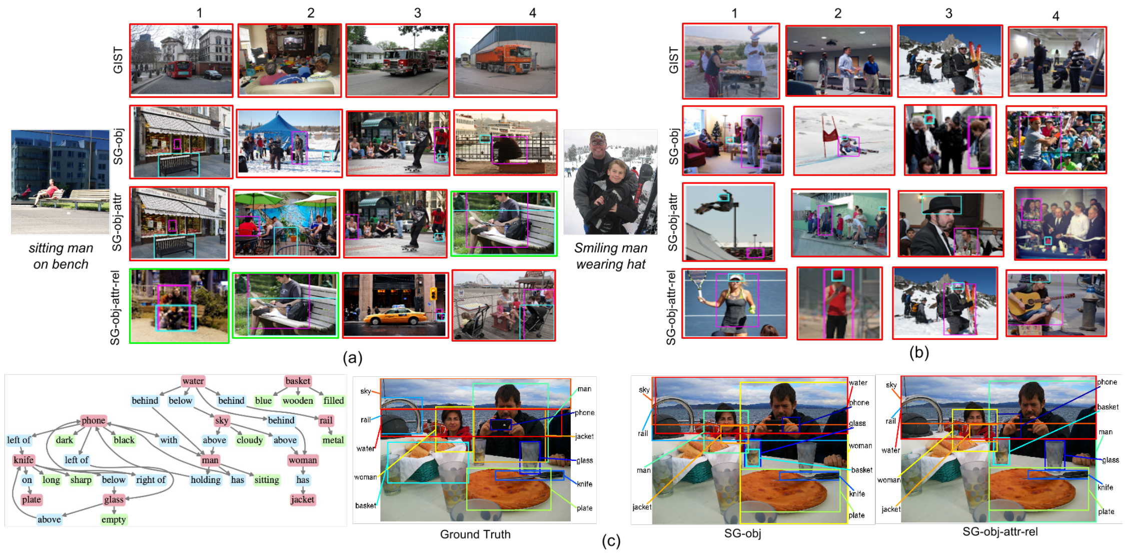


Figure 7.8: Top-*4* retrieval results returned by different methods using two different partial scene graph queries (a, b). Differences in fully automatic scene graph grounding when applying these methods to a particular test image (c).

query image I_q. We repeat this process using 150 randomly selected images from our test set and evaluate the ranking of the query image over all 150 trials. Table 7.2 (a) gives the results in the form of recall at rank k (higher is better) and median rank of the query image I_q (lower is better). Fig. 7.7 (a) plots the recall over k. Note that the GIST, SIFT, and CNN baselines are meaningless here, as they would always rank the query image highest.

Results. In Table 7.2 (a), we observe that the detailed semantics encoded in our scene graphs greatly increases the performance for entire scene retrieval. Compared to SG-obj , SG-obj-attr-rel decreases the median rank of the query image from 28 by half to 14. Recall at k shows similar results, where SG-obj-attr-rel increases recall over SG-obj by 2% (R@1), 4.7% (R@5), and 8.6% (R@10), respectively. This performance improvement increases for larger values of k (Fig.7.7 (a), blue vs red curve), to around 15% at 30. SG-obj-attr outperforms SG-obj , indicating that attributes are useful, and is in turn outperformed by SG-obj-attr-rel .

Fig. 7.6 shows corresponding qualitative results: on the left, our full model successfully identifies and localizes the "old woman with a jacket, sitting next to a sitting man with a striped tie". On the right, even though some objects are misplaced, SG-obj-attr-rel is able to correctly place the "dark blanket on the bed", while SG-obj-attr incorrectly grounds the blanket to a dark region of the test image.

Small scene graph queries

We have shown that our model is able to handle the complexities of full scene graph queries. Here we show that our model can also be used to retrieve meaningful results given simpler, more human-interpretable scene graph queries.

Specifically, we mine our dataset for re-occurring scene subgraphs containing two objects, one relationship, and one or two attributes, such as "sitting man on bench" and "smiling man wearing hat." We retain only subgraphs that occur at least 5 times in our test set, resulting in 598 scene subgraphs. We randomly selected 119 to be used as queries. For each subgraph query, we find the set of test images $I_1, \ldots, I_\ell$ that include it. We hold out I_1 from the test set and use the subgraph to rank the remaining 999 test images.

For the baseline methods we use the image I_1 rather than the graph to rank the test images. For the SIFT baseline, for each SIFT descriptor from I_1, we compute its 100 nearest neighbors among the descriptors from all other test images, and sort the test images by the number of these neighbors they contain. For the GIST and CNN baselines, we rank the test images based on the L_2 distance between the descriptor for I_1 and the descriptor for the test image.

We adapt the metrics from [85]; specifically, for each method we report the median rank of the highest ranked true positive image $I_2, \ldots, I_\ell$ across all queries and the recall at k for various values of k. Results are shown in Table 7.2 (b) and Fig. 7.7 (b). Examples of retrieved images can be seen in Fig. 7.8 (a,b), for the two queries mentioned above.

Results. In Table 7.2 (b), we see that our full model SG-obj-attr-rel again outperforms SG-obj by a large margin, reducing the median rank from 17 to 11. SG-obj-attr comes close, with a median rank of 12. In terms of recall at k, SG-obj-attr-rel again dominates, improving over SG-obj by 5% (R@1), 3.4% (R@5), and 6.7% (R@10), respectively. This gap increases up to around 12% at 40 (Fig. 7.7 (b)).

Qualitatively, Fig. 7.8 (a) shows that SG-obj-attr-rel retrieves correct results for "sitting man on bench." In comparison, the first result returned by SG-obj and SG-obj-attr contains both a man and a bench that are correctly localized, but the man is not sitting on the bench. Fig. 7.8 (b) shows that although SG-obj-attr-rel retrieves incorrect results for "smiling man wearing hat", it fails gracefully by returning images of a smiling woman wearing a hat, a man wearing a hat who is not smiling, and a smiling man wearing a helmet.

Object localization

We have shown that our scene graph representation improves image retrieval results; here, we show that it can also aid in localizing individual objects. For each image I and its corresponding ground-truth (GT) scene graph G from our test set, we use each model to generate a grounding of G to I.

For each object in G, we compute the intersection over union (IoU) between its GT position in the image and its position in the grounding generated by our model.

Fig. 7.8 (c) gives an example scene graph and its computed grounding under SG-obj-attr-rel and SG-obj . SG-obj labels "man" as "woman" and vice versa, but SG-obj-attr-rel is able to correct this error. Note that the scene graph does not specify any direct relationships between the man and the woman; instead, SG-obj-attr-rel must rely on the relationships between the two people and other objects in the scene ("man holding phone", "woman has jacket").

To quantitatively compare our models, we report the median IoU across all object instances in all test images, (Med IoU) and the fraction of object instances with IoU above various thresholds (IoU@t). Table 7.2 shows that SG-obj-attr-rel outperforms both SG-obj and SG-obj-attr on all metrics. Interestingly, comparing SG-obj-attr-rel to SG-obj shows a nearly five-fold increase in median IoU. The generally low values for median IoU highlight the difficulty of the automatic scene graph grounding task. Fig. 7.7 (c) shows that SG-obj-attr-rel performs particularly well compared to the baseline models at IoU thresholds below 0.5.

To gain more insight into the performance of our model, for each object instance in the test set we compute the difference in IoU with the GT between the grounding found by SG-obj-attr-rel and the grounding found by SG-obj . For each object class that occurs at least 4 times in the test set, we compute the median difference in IoU between the two methods, and perform the same analysis for (object, relationship, object) tuples, where for each such tuple in the test set we compute the mean IoU of the objects referenced by the tuple. The top and bottom 5 object classes and tuples are visualized in Fig. 7.7 (d).

This figure suggests that the context provided by SG-obj-attr-rel helps to localize rare objects (such as "van" and "guy") and objects with large appearance variations (such as "pillow" and "building"). SG-obj-attr-rel also gives large gains for tuples with well-defined spatial constraints ("jacket on woman", "chair under man"). Tuples encoding less well-defined spatial relationships ("man under sky", "girl on snow") may suffer in performance as the model penalizes valid configurations not seen at training time.

7.1.7 Conclusion

In this Chapter, we have used scene graphs as a novel representation for detailed semantics in visual scenes, and introduced a novel dataset of scene graphs grounded to real-world images. We have used this representation and dataset to construct a CRF model for semantic image retrieval using scene graphs as queries. We have shown that this model outperforms methods based on object detection and low-level visual features. We believe that semantic image retrieval is one of many exciting applications of our scene graph representation and dataset, and hope that more will follow.

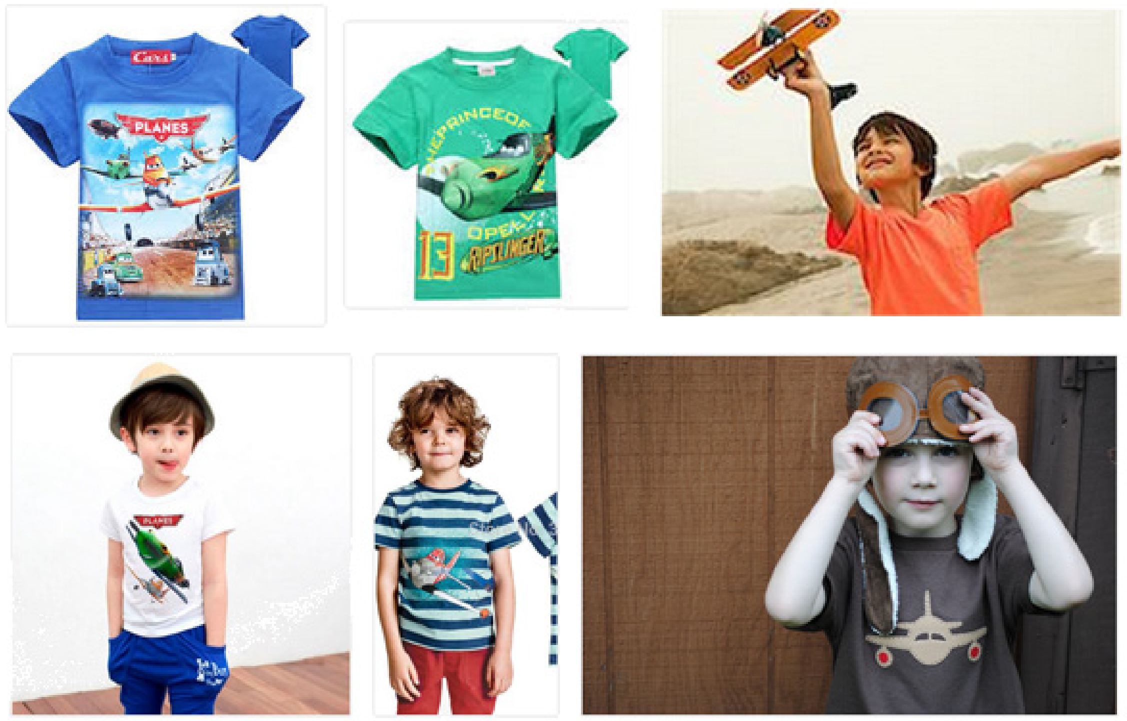


Figure 7.9: Actual results using a popular image search engine (top row) and ideal results (bottom row) for the query *a boy wearing a t-shirt with a plane on it.*

7.2 Image Retrieval using Language Queries

7.2.1 Introduction

One of the big remaining challenges in image retrieval is to be able to search for very specific images. The continuously growing number of images that are available on the web gives users access to almost any picture they can imagine, but in order to find these images users have to be able to express what they are looking for in a detailed and efficient way. For example, if a user wants to find an image of *a boy wearing a t-shirt with a plane on it*, an image retrieval system has to understand that the image should contain a boy who is wearing a shirt and that on that shirt is a picture of a plane.

Keyword-based image retrieval systems are clearly unable to deal with the rich semantics of such a query [140]. They might be able to retrieve images that contain a boy, a t-shirt and a plane but they are unable to interpret the relationships and attributes of these objects which is crucial for retrieving the correct images. As shown in Figure 7.9, a possible but incorrect combination of these objects is that a boy is wearing a t-shirt and playing with a toy plane.

One proposed solution to these issues is the mapping of image descriptions to multi-modal embeddings of sentences and images and using these embeddings to retrieve images [180, 106, 109, 149, 34].

However, one problem of these models is that they are trained on single-sentence captions which are typically unable to capture the rich content of visual scenes in their entirety. Further, the coverage of the description highly depends on the subjectivity of human perception [195]. Certain details such as whether there is a plane on the boy's shirt or not might seem irrelevant to the person who writes the caption, but for another user this difference might determine whether a result is useful or not.

[99] try to solve these problems by annotating images with a graph-based semantic representation called a *scene graph* which explicitly captures the objects in an image, their attributes and the relations between objects. They plausibly argue that paragraph-long image descriptions written in natural language are currently too complex to be mapped automatically to images and instead they show that very detailed image descriptions in the form of scene graphs can be obtained via crowdsourcing. They also show that they can perform semantic image retrieval on unannotated images using partial scene graphs.

However, one big shortcoming of their model is that it requires the user to enter a query in the form of a scene graph instead of an image description in natural language which is unlikely to find widespread adoption among potential users. To address this problem, we propose a new task of parsing image descriptions to scene graphs which can then be used as a query for image retrieval.

While our main goal is to show the effectiveness of parsing image descriptions for image retrieval, we believe that scene graphs can be a useful intermediate representation for many applications that involve text and images. One great advantage of such an intermediate representation is the resulting modularity which allows independent development, improvement and reuse of NLP, vision and graphics subsystems. For example, we can reuse a scene graph parser for systems that generate 2D-scenes [273] or 3D-scenes [23] which require input in the form of similar graph-based representations to which a scene graph can be easily converted.

In this chapter, we introduce the task of parsing image descriptions to scene graphs. We build and evaluate a rule-based and a classifier-based scene graph parser which map from dependency syntax representations to scene graphs. We use these parsers in a pipeline which first parses an image description to a scene graph and then uses this scene graph as input to a retrieval system. We show that such a pipeline outperforms a system which only considers objects in the description and we show that the output of both of our parsers is almost as effective as human-constructed scene graphs in retrieving images. Lastly, we demonstrate the more general applicability of our parsers by generating 3D scenes from their output.

We make our parsers and models available at http://nlp.stanford.edu/software/scenegraph-parser.shtml.

7.2.2 Task Description

Our overall task is retrieving images from image descriptions which we split into two sub-tasks: Parsing the description to scene graphs and retrieving images with scene graphs. In this chapter, we focus exclusively on the first task. For the latter, we use a reimplementation of the system by [99] which we briefly describe in the next section.

Image Retrieval System

The image retrieval system by [99] is based on a conditional random field (CRF) [121] model which – unlike the typical CRFs in NLP – is not a chain model but instead capturing image region proximity. This model ranks images based on how likely it is that a given scene graph is grounded to them. The model first identifies potential object regions in the image and then computes the most likely assignment of objects to regions considering the classes of the objects, their attributes and their relations. The likelihood of a scene graph being grounded to an image is then approximated as the likelihood of the most likely assignment of objects to regions.

Parsing to Scene Graphs

The task of parsing image descriptions to scene graphs is defined as following. Given a set of object classes C, a set of relation types R, a set of attribute types A, and a sentence S we want to parse S to a scene graph $G = (O, E)$. $O = \{o_1, ..., o_n\}$ is a set of objects mentioned in S and each o_i is a pair (c_i, A_i) where $c_i \in C$ is the class of o_i and $A_i \subseteq A$ are the attributes of o_i. $E \subseteq O \times R \times O$ is the set of relations between two objects in the graph. For example, given the sentence $S =$ *"A man is looking at his black watch"* we want to extract the two objects $o_1 = (man, \emptyset)$ and $o_2 = (watch, \{black\})$, and the relations $e_1 = (o_1, look\ at, o_2)$ and $e_2 = (o_1, have, o_2)$. The sets C, R and A consist of all the classes and types which are present in the training data.

Data

We reuse a dataset which we collected for a different task using Amazon Mechanical Turk (AMT) in a similar manner as [99] and [180]. We originally annotated 4,999 images from the intersection of the YFCC100m [234] and Microsoft COCO [139] datasets. However, unlike previous work, we split the process into two separate passes with the goal of increasing the number of objects and relations per image.

In the first pass, AMT workers were shown an image and asked to write a one sentence description of the entire image or any part of it. To get diverse descriptions, workers were shown the previous descriptions written by other workers for the same image and were asked to describe something about the image which had not been described by anyone else. We ensured diversity in sentence descriptions by a real-time BLEU score [171] threshold between a new sentence and all the previous ones.

In the second pass, workers were presented again with an image and with one of its sentences. They were asked to draw bounding boxes around all the objects in the image which were mentioned in the sentence and to describe their attributes and the relations between them. This step was

	Raw	**Processed**	**Filtered**
Images	4,999	4,999	4,524
Sentences	88,188	88,188	50,448
Sentences per image	17.6	17.6	11.2
Object classes	18,515	15,734	798
Attribute types	7,348	6,442	277
Relation types	9,274	7,507	131
Objects per image	21.2	21.2	14.6
Attributes per image	16.2	16.4	10.7
Relations per image	18.6	18.6	10.3
Attributes per sent.	0.92	0.93	0.93
Relations per sent.	1.06	1.06	0.96

Table 7.3: Aggregate statistics of the raw, canonicalized (processed) and filtered datasets.

repeated for each sentence of an image and finally the partial scene graphs are combined to one large scene graph for each image. While the main purpose of the two-pass data collection was to increase the number of objects and relations per image, it also provides as a byproduct a mapping between sentences and partial scene graphs which gives us a corpus of sentence-scene graph pairs that we can use to train a parser.

Preprocessing The AMT workers were allowed to use any label for objects, relations and attributes and consequently there is a lot of variation in the data. We perform several preprocessing steps to canonicalize the data. First, we remove leading and trailing articles from all labels. Then we replace all the words in the labels with their lemmata and finally we split all attributes with a conjunction such as *red and green* into two individual attributes.

We also follow [99] and discard all objects, relations and attributes whose class or type appears less than 30 times in the entire dataset for the following two reasons. First and foremost, computer vision systems require multiple training examples for each class and type to be able to learn useful generalizations, and second, rare classes and types are often a result of AMT workers making mistakes or not understanding the task properly. As we make the assumption that the scene graph of one sentence is complete, i.e., that it captures all the information of the sentence, we have to apply a more aggressive filtering which discards the entire scene graph of a sentence in case one of its objects, attributes or relations is discarded due to the threshold. In case we discard all sentences of an image, we discard the entire image from our data. Despite the aggressive filtering, the average number of objects, relations and attributes per image only drops by 30-45% and we only discard around 9% of the images (see Table 7.3).

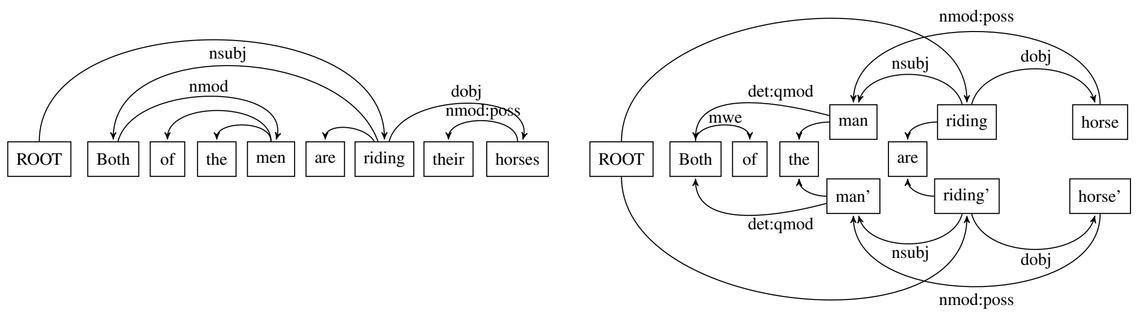


Figure 7.10: Dependency tree and final semantic graph of a sentence. *men* is promoted to be the subject; *men*, *riding*, and *horses* are duplicated; and *their* is deleted following coreference resolution.

7.2.3 Scene Graph Parsers

We implement two parsers: a rule-based parser and a classifier-based parser. Both of our parsers operate on a linguistic representation which we refer to as a *semantic graph*. We obtain semantic graphs by parsing the image descriptions to dependency trees followed by several tree transformations. In this section, we first describe these tree transformations and then explain how our two parsers translate the semantic graph to a scene graph.

Semantic Graphs

A Universal Dependencies [39] parse is in many ways close to a shallow semantic representation and therefore a good starting point for parsing image descriptions to scene graphs. Basic dependency trees, however, tend to follow the linguistic structure of sentences too closely which requires some post-processing of the parses to make them more useful for a semantic task. We start with the *enhanced* dependency representation output by the Stanford Parser v3.5.2 [111][2] and then perform three additional processing steps to deal with complex quantificational modifiers, to resolve pronouns and to handle plural nouns.

Quantificational modifiers Several common expressions with light nouns such as *a lot of* or *a dozen of* semantically act like quantificational determiners [215]. From a syntactic point of view, however, these expressions are the head of the following noun phrase. While one of the principles of the Universal Dependencies representation is the primacy of content words [39], light nouns are treated like any other noun. To make our dependency trees better suited for semantic tasks, we change the structure of all light noun expressions from a manually compiled list. We make the first word the head of all the other words in the expression and then make this new multi-word expression

[2]We augment the parser's training data with the Brown corpus [152] to improve its performance on image descriptions which are often very different from sentences found in newswire corpora.

a dependent of the following noun phrase. This step guarantees that the semantic graph for *both cars* and for *both of the cars* have similar structures in which the semantically salient word *cars* is the head.

Pronoun resolution Some image descriptions such as *"a bed with a pillow on it"* contain personal pronouns. To recover all the relations between objects in this sentence it is crucial to know that *it* refers to the object *a bed* and therefore we try to resolve all pronouns. We found in practice that document-level coreference systems (e.g. [131]) were too conservative in resolving pronouns and hence we implement an intrasential pronoun resolver inspired by the first three rules of the Hobbs algorithm [83] which we modified to operate on dependency trees instead of constituency trees. We evaluate this method using 200 randomly selected image descriptions containing pronouns. Our pronoun resolver has an accuracy of 88.5% which is significantly higher than the accuracy of 52.8% achieved by the coreference system of [131].

Plural nouns Plural nouns are known to be a major challenge in semantics in general [166], and also in our task. One particular theoretical issue is the collective-distributive ambiguity of sentences with multiple plural nouns. For example, to obtain the intended distributive reading of *"three men are wearing jeans"* we have to extract three *man* objects and three *jeans* objects and we have to connect each *man* object to a different *jeans* object. On the other hand, to get the correct parse of *"three men are carrying a piano"* we probably want to consider the collective reading and extract only one *piano* object. A perfect model thus requires a lot of world knowledge. In practice, however, the distributive reading seems to be far more common so we only consider this case.

To make the dependency graph more similar to scene graphs, we copy individual nodes of the graph according to the value of their numeric modifier. We limit the number of copies per node to 20 as our data only contains scene graphs with less than 20 objects of the same class. In case a plural noun lacks such a modifier we make exactly one copy of the node.

Figure 7.10 shows the original dependency tree and the final semantic graph for the sentence "*Both of the men are riding their horses*".

Rule-Based Parser

Our rule-based parser extracts objects, relations and attributes directly from the semantic graph. We define in total nine dependency patterns using Semgrex[3] expressions. These patterns capture the following constructions and phenomena:

- Adjectival modifiers

[3]http://nlp.stanford.edu/software/tregex.shtml

- Subject-predicate-object constructions and subject-predicate constructions without an object
- Copular constructions
- Prepositional phrases
- Possessive constructions
- Passive constructions
- Clausal modifiers of nouns

With the exception of possessives for which we manually add a *have* relation, all objects, relations and attributes are words from the semantic graph. For example, for the semantic graph in Figure 7.10, the *subject-predicate-object* pattern matches $man \xleftarrow{nsubj} riding \xrightarrow{dobj} horse$ and $man' \xleftarrow{nsubj} riding' \xrightarrow{dobj} horse'$. From these matches we extract two *man* and two *horse* objects and add *ride* relations to the two *man-horse* pairs. Further, the *possesive* pattern matches $man \xleftarrow{nmod:poss} horse$ and $man' \xleftarrow{nmod:poss} horse'$ and we add *have* relations to the two *man-horse* pairs.

Classifier-Based Parser

Our classifier-based parser consists of two components. First, we extract all candidate objects and attributes, and second we predict relations between objects and the attributes of all objects.

Object and Attribute Extraction We use the semantic graph to extract all object and attribute candidates. In a first step we extract all nouns, all adjectives and all intransitive verbs from the semantic graph. As this does not guarantee that the extracted objects and attributes belong to known object classes or attribute types and as our image retrieval model can only make use of known classes and types, we predict for each noun the most likely object class and for each adjective and intransitive verb the most likely attribute type. To predict classes and types, we use an L_2-regularized maximum entropy classifier which uses the original word, the lemma and the 100-dimensional GloVe word vector [176] as features.

Relation Prediction The last step of the parsing pipeline is to determine the attributes of each object and the relations between objects. We consider both of these tasks as a pairwise classification task. For each pair (x_1, x_2) where x_1 is an object and x_2 is an object or an attribute we predict the relation y which can be any relation seen in the training data, or one of the two special relations *IS* and *NONE* which indicate that x_2 is an attribute of x_1 or no relation exists, respectively. We noticed that for most pairs for which a relation exists, x_1 and x_2 are in the same constituent, i.e. their lowest common ancestor is either one of the two objects or a word in between them. We therefore consider only pairs which satisfy this constraint to improve precision and to limit the number of predictions.

For the predictions, we use again an L_2-regularized maximum entropy classifier with the following features:

Object features The original word and lemma, and the predicted class or type of x_1 and x_2.

Lexicalized features The word and lemma of each token between x_1 and x_2. If x_1 or x_2 appear more than once in the sentence because they replace a pronoun, we only consider the words in between the closest mentions of x_1 and x_2.

Syntactic features The concatenated labels (i.e., syntactic relation names) of the edges in the shortest path from x_1 to x_2 in the semantic graph.

We only include objects in the scene graph which have at least one attribute or which are involved in at least one relation. The idea behind that is to prevent very abstract nouns such as *setting* or *right* to be part of the scene graph which are typically not part of relations. However, we observed for around 30% of the sentences in the development set that the parser did not extract any relations or attributes from a sentence which resulted in an empty scene graph. In these cases, we include all candidate objects in the scene graph.

Training As the scene graph's objects and attributes are not aligned to the sentence, we have to align them in an unsupervised manner. For each sentence, we extract object and attribute candidates from the semantic graph. For each object-relation-object triple or object-attribute pair in the scene graph we try to align all objects and attributes to a candidate by first checking for exact string match of the word or the lemma, then by looking for candidates within an edit distance of two, and finally by mapping the object or attribute and all the candidates to 100-dimensional GloVe word vectors and picking the candidate with the smallest euclidean distance. To limit the number of false alignments caused by annotators including objects in the scene graph that are not present in the corresponding sentence, we also compute the euclidean distances to all the other words in the sentence and if the closest match is not in the candidate set we discard the training example.

We use this data to train both of our classifiers. For the object and attribute classifier, we only consider the alignments between words in the description and objects or attributes in the graph.

For the relation predictor, we consider the complete object-relation-object and object-*is*-attribute triples. All the aligned triples constitute our positive training examples for a sentence. For all the object-object and object-attribute pairs without a relation in a sentence, we generate negative examples by assigning them a special $NONE$ relation. We sample from the set of $NONE$ triples to have the same number of positive and negative training examples.

	Train	**Dev**	**Test**
Images	3,614	454	456
Sentences	40,315	4,953	5,180
Relation instances	38,617	4,826	4,963
Attribute instances	37,580	4,644	4,588

Table 7.4: Aggregate statistics of the training, development (dev) and test sets.

7.2.4 Experiments

For our experiments, we split the data into training, development and held-out test sets of size 3,614, 454, and 456 images, respectively. Table 7.4 shows the aggregated statistics of our training and test sets. We compare our two parsers against the following two baselines.

Nearest neighbor Our first baseline computes a term-frequency vector for an input sentence and returns the scene graph of the nearest neighbor in the training data.

Object only Our second baseline is a parser that only outputs objects but no attributes or relationships. It uses the first two components of the classifier-based parser, namely the semantic graph processor and the object extractor, and then simply outputs all candidate objects.

We use the downstream performance on the image retrieval task as our main evaluation metric. We train our reimplementation of the model by [99] on our training set with human-constructed scene graphs. For each sentence we use the parser's output as a query and rank all images in the test set. For evaluation, we consider the human-constructed scene graph G_h of the sentence and construct a set of images $I = i_1, ..., i_n$ such that G_h is a subgraph of the image's complete scene graph. We compute the rank of each image in I and compute recall at 5 and 10 based on these ranks[4]. We also compute the median rank of the first correct result. We compare these numbers against an oracle system which uses the human-constructed scene graphs as queries instead of the scene graphs generated by the parser.

One drawback of evaluating on a downstream task is that evaluation is typically slower compared to using an intrinsic metric. We therefore also compare the parsed scene graphs to the human-constructed scene graphs. As scene graphs consist of object instances, attributes, and relations and are therefore similar to Abstract Meaning Representation (AMR) [5] graphs, we use Smatch F1 [20] as an additional intrinsic metric.

[4]As in [99], we observed that the results for recall at 1 were very unstable so we only report recall at 5 and 10 which are typically also more relevant for real-world systems that return multiple results.

	Development set				Test set			
	Smatch	**R@5**	**R@10**	**Med. rank**	**Smatch**	**R@5**	**R@10**	**Med. rank**
Nearest neighbor	32%	1.2%	2.3%	206	32%	1.1%	2.3%	205
Object only	**48%**	15.0%	29.3%	20	**48%**	12.6%	24.8%	25
Rule	43%	16.4%	31.6%	17	44%	13.5%	**27.1%**	**20**
Classifier	47%	**16.7%**	**32.9%**	**16**	47%	**13.8%**	**27.1%**	**20**
Oracle	-	19.4%	39.8%	13	-	16.6%	33.4%	15

Table 7.5: Intrinsic (Smatch F1) and extrinsic (recall at 5 and 10, and median rank) performance of our two baselines, our rule-based and our classifier-based parser.

	R@5	**R@10**	**Med. rank**
[99]	30.3%	47.9%	11
Our implementation	27.6%	45.6%	12

Table 7.6: Comparison of the results of the original implementation by [99] and our implementation. Both systems were trained and tested on the data sets of the original authors.

7.2.5 Results and Discussion

Table 7.5 shows the performance of our baselines and our two final parsers on the development and held-out test set.

Oracle results Compared to the results of [99], the results of our oracle systems are significantly worse. To verify the correctness of our implementation, the original authors provided us with their training and test set. Table 7.6 shows that our reimplementation performs almost as well as their original implementation. We hypothesize that there are two main reasons for the drop in performance when we train and evaluate our system on our dataset. First, our dataset is a lot more diverse and contains many more object classes and relation and attribute types. Second, the original authors only use the most common queries for which there exist at least five results to retrieve images while we evaluate on all queries.

Effectiveness of Smatch F1 As mentioned in the previous section, having an intrinsic evaluation metric can reduce the length of development cycles compared to using only an extrinsic evaluation. We hoped that Smatch F1 would be an appropriate metric for our task but our results indicate that there is no strong correlation between Smatch F1 and the performance of the downstream task.

Comparison of rule-based and classifier-based system In terms of image retrieval performance, there does not seem to be a significant difference between our rule-based system and our classifier-based system. On the development set the classifier-based system slightly outperforms the rule-based system but on the test set both seem to work equally well. Nevertheless, their results

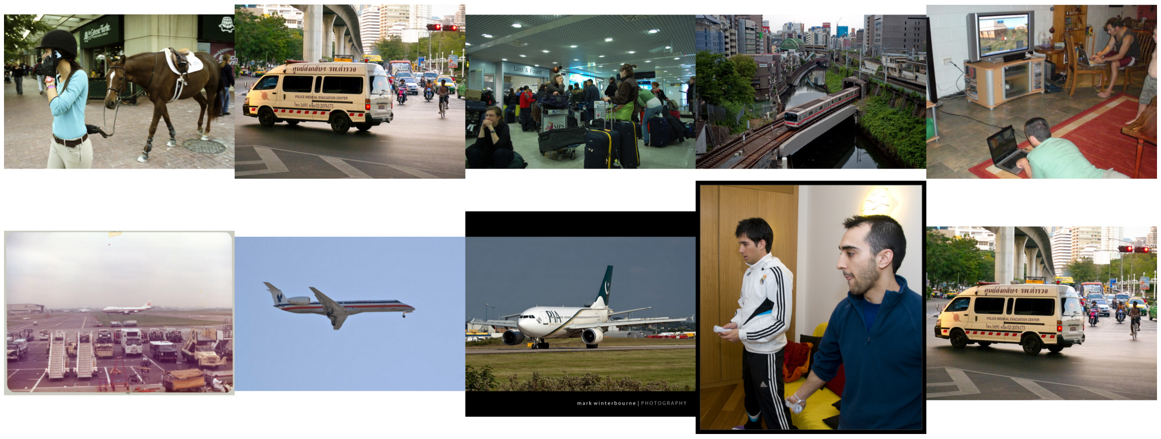


Figure 7.11: Top 5 results of the object only baseline (top row) and our classifier-based parser (bottom row) for the query *"The white plane has one blue stripe and one red stripe"*. The *object only* system seems to be mainly concerned with finding images that contain two *stripe* objects at the expense of finding an actual plane. Our classifier-based parser also outputs the relation between the stripes and the plane and the colors of the stripes which helps the image retrieval system to return the correct results.

differ in some cases. One strength of the classifier-based system is that it learns that some adjectival modifiers like *several* should not be attributes. It is also able to learn some basic implications such as *the shirt looks dirty* implies in the context of an image that the shirt is dirty. On the other hand, the rule-based system tends to be more stable in terms of extracting relations while the classifier-based system more often only extracts objects from a sentence.

Comparison to baselines As shown in Table 7.5, both of our parsers outperform all our baselines in terms of recall at 5 and 10, and the median rank. This difference is particularly significant compared to the *nearest neighbor* baseline which confirms the complexity of our dataset and shows that it is not sufficient to simply memorize the training data.

The *object only* baseline is a lot stronger but still performs consistently worse than our two parsers. To understand in what ways our parsers are superior to the *object only* baseline, we performed a qualitative analysis. A comparison of the results reveals that the image retrieval model is able to make use of the extracted relations and attributes. Figure 7.11 shows the top 5 results of our classifier-based parser and the *object only* baseline for the query *"The white plane has one blue stripe and one red stripe"*. While the *object only* model seems to be mainly concerned with finding good matches for the two *stripe* objects, the output of our parser successfully captures the relation between the plane and the stripes and correctly ranks the two planes with the blue and red stripes as the top results.

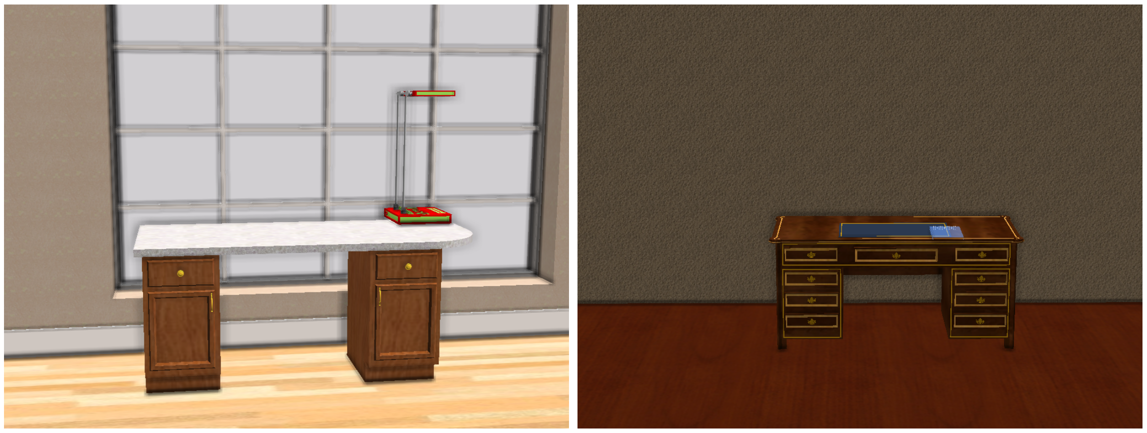


Figure 7.12: 3D scenes for the sentences *"There is a wooden desk with a red and green lamp on it"* and *"There is a desk with a notepad on it"*.

Error analysis The performance of both of our parsers comes close to the performance of the oracle system but nevertheless there still remains a consistent gap. One of the reasons for the lower performance is that some human-constructed scene graphs contain information which is not present in the description. The human annotators saw both the description and the image and could therefore generate scene graphs with additional information.

Apart from that, we find that many errors occur with sentences which require some external knowledge. For example, our parser is not able to infer that *"a woman in black"* means that a woman is wearing black clothes. Likewise it is not able to infer that *"a jockey is wearing a green shirt and matching helmet"* implies that he is wearing a green helmet.

Other errors occur in some sentences which talk about textures. For example, our parsers assume that "a dress with polka dots" implies that there is a relation between one *dress* object and multiple *polka dot* objects instead of inferring that there is one *dress* object with the attribute *polka-dotted*.

One further source of errors are wrong dependency parses. Both of our parsers heavily rely on correct dependency parses and while making the parser's training data more diverse did improve results, we still observe some cases where sentences are parsed incorrectly leading to incorrect scene graphs.

7.2.6 Other Tasks

As mentioned before, one appeal of parsing sentences to an intermediate representation is that we can also use our parser for other tasks that make use of similar representations. One of these tasks is generating 3D scenes from textual descriptions [23]. Without performing any further modifications, we replaced their parser with our classifier-based parser and used the resulting system to generate 3D scenes from several indoor scene descriptions. Two of these generated scenes are shown in Figure

7.12. Our impression is that the system performs roughly equally well using this parser compared to the one used in the original work.

7.2.7 Related Work

Image retrieval Image retrieval is one of the most active areas in computer vision research. Very early work mainly focused on retrieving images based on textual descriptions, while later work focused more on content-based image retrieval systems which perform retrieval directly based on image features. [195], [140], and [214] provide overviews of the developments of this field over the last twenty years. Most of this work focused on retrieving images from keywords which are not able to capture many semantic phenomena as well as natural language or our scene graph representation can.

Multi-modal embeddings Recently, multi-modal embeddings of natural language and images got a lot of attention [224, 106, 180, 109, 149, 34]. These embeddings can be used to retrieve images from captions and generating captions from images. As mentioned in the introduction, these models are trained on single-sentence image descriptions which typically cannot capture all the details of a visual scene. Further, unlike our modular system, they cannot be used for other tasks that require an interpretable semantic representation.

Parsing to graph-based representations Representing semantic information with graphs has recently experienced a resurgence caused by the development of the Abstract Meaning Representation (AMR) [5] which was followed by several works on parsing natural language sentences to AMR [61, 247, 250]. Considering that AMR graphs are, like dependency trees, very similar to scene graphs, we could have also used this representation and transformed it to scene graphs. However, the performance of AMR parsers is still not competitive with the performance of dependency parsers which makes dependency trees are more stable starting point.

There also exists some prior work on parsing scene descriptions to semantic representations. As mentioned above, [23] present a rule-based system to parse natural language descriptions to *scene templates*, a similar graph-based semantic representation. [45] parse image descriptions to a dependency grammar representation which they also use for image retrieval. [138] also use rules to transform dependency trees into semantic graphs which they use for video search. All of this work, however, only consider a limited set of relations while our approach can learn an arbitrary number of relations. Further, they all exclusively use very specific rule-based systems whereas we also introduced a more general purposed classifier-based parser.

7.2.8 Conclusion

We presented two parsers which can translate image descriptions to scene graphs. We showed that their output is almost as effective for retrieving images as human-generated scene graphs and that including relations and attributes in queries outperforms a model which only considers objects. We also demonstrated that our parser is well suited for other tasks which require a semantic representation of a visual scene.

Chapter 8

Conclusions and Future Directions

8.1 Conclusion

The core theme throughout this thesis has been to demonstrate the utility of crowdsourced visual knowledge representations. We started off by first introducing and exploring the Visual Genome dataset. Visual Genome provides a multi-layered understanding of pictures. It allows for a multi-perspective study of an image, from pixel-level information like objects, to relationships that require further inference, and to even deeper cognitive tasks like question answering. It is a comprehensive dataset for training and benchmarking the next generation of computer vision models.

Next, we showcased how different crowdsourcing techniques were used to collect the Visual Genome Dataset. We specifically honed in improving binary labeling tasks. We presented a technique that produces extremely rapid judgments for binary and categorical labels. We demonstrate that it is possible to rectify worker errors on a breadth of common labeling tasks such as image verification, word similarity, sentiment analysis and topic classification.

Once the data was collected, we studied automatic methods to extract structured representations like visual relationships without any human intervention. We built a model that leveraged existing language models to help predict likely relationships between objects in an image. Finally, we used our dataset to train a model that can improve semantic image retrieval by mapping language queries first to scene graphs and then to images.

With Visual Genome, we expect these models to develop a broader understanding of our visual world, complementing computers' capacities to detect objects with abilities to describe those objects and explain their interactions and relationships. Visual Genome is a large *formalized knowledge representation* for visual understanding and a more *complete set of descriptions and question answers* that *grounds visual concepts to language.*

8.2 Future Directions and Applications

We have explored the individual components of the Visual Genome dataset and presented experiments with results for tasks such as attribute classification, relationship classification, description generation, and question answering. We have also developed models for relationship extraction and semantic image retrieval. There are, however, more applications and experiments for which our dataset can be used. In this section, we note a few potential applications that Visual Genome can enable.

8.2.1 Dense image captioning.

We have seen numerous image captioning papers [108, 150, 106, 242] that attempt to describe an entire image with a single caption. However, these captions do not exhaustively describe every part of the scene. An natural extension to this application, which the Visual Genome dataset enables, is the ability to create dense captioning models that describe parts of the scene.

8.2.2 Visual question answering.

While visual question answering has been studied as a standalone task [263, 190, 3, 66], we introduce a dataset that combines all of our question answers with descriptions and scene graphs. Future work can build supervised models that utilize various components of Visual Genome to tackle question answering.

8.2.3 Image understanding.

While we have seen a surge of image captioning [108] and question answering [3] models, there has been little work on creating more comprehensive evaluation metrics to measure how well these models are performing. Such models are usually evaluated using BLEU, CIDEr, or METEOR and other similar metrics that do not effectively measure how well these models understand the image [25]. The Visual Genome scene graphs can be used as a measurement for image understanding. Generated descriptions and answers can be matched against the ground truth scene graph of an image to evaluate its corresponding model.

8.2.4 Relationship extraction.

Relationship extraction has been extensively studied in information retrieval and natural language processing [268, 77, 37, 223]. Visual Genome is the first large-scale visual relationship dataset. This dataset can be used to study the extraction of visual relationships [202] from images, and its interactions between objects can also be used to study action recognition [257, 187] and spatial orientation between objects [79, 183].

8.2.5 Semantic image retrieval.

Previous work has already shown that scene graphs can be used to improve semantic image search [100, 209]. Further methods can be explored using our region descriptions combined with region graphs. Attention-based search methods can also be explored where the area of interest specified by a query is also localized in the retrieved images.

Bibliography

[1] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. Measuring the objectness of image windows. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2189–2202, 2012.

[2] Bjoern Andres, Thorsten Beier, and Jörg H Kappes. OpenGM: A C++ library for discrete graphical models. *arXiv preprint arXiv:1206.0111*, 2012.

[3] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. *arXiv preprint arXiv:1505.00468*, 2015.

[4] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 86–90, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics.

[5] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, 2013.

[6] Alexander C Berg, Tamara L Berg, Hal Daume III, Jesse Dodge, Amit Goyal, Xufeng Han, Alyssa Mensch, Margaret Mitchell, Aneesh Sood, Karl Stratos, et al. Understanding and predicting importance in images. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3562–3569. IEEE, 2012.

[7] Tamara L Berg, Alexander C Berg, and Jonathan Shih. Automatic attribute discovery and characterization from noisy web data. In *Computer Vision–ECCV 2010*, pages 663–676. Springer, 2010.

[8] Alessandro Bergamo, Lorenzo Torresani, and Andrew Fitzgibbon. Picodes: Learning a compact code for novel-category recognition. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N.

Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2088–2096. 2011.

[9] Michael S Bernstein, Joel Brandt, Robert C Miller, and David R Karger. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 33–42. ACM, 2011.

[10] Michael S Bernstein, Greg Little, Robert C Miller, Björn Hartmann, Mark S Ackerman, David R Karger, David Crowell, and Katrina Panovich. Soylent: a word processor with a crowd inside. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 313–322. ACM, 2010.

[11] Justin Betteridge, Andrew Carlson, Sue Ann Hong, Estevam R Hruschka Jr, Edith LM Law, Tom M Mitchell, and Sophie H Wang. Toward never ending language learning. In *AAAI Spring Symposium: Learning by Reading and Learning to Read*, pages 1–2, 2009.

[12] Arijit Biswas and Devi Parikh. Simultaneous active learning of classifiers & attributes via relative feedback. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 644–651. IEEE, 2013.

[13] Jonathan Bragg, Mausam Daniel, and Daniel S Weld. Crowdsourcing multi-label classification for taxonomy creation. In *First AAAI conference on human computation and crowdsourcing*, 2013.

[14] Steve Branson, Kristjan Eldjarn Hjorleifsson, and Pietro Perona. Active annotation translation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 3702–3709. IEEE, 2014.

[15] Steve Branson, Catherine Wah, Florian Schroff, Boris Babenko, Peter Welinder, Pietro Perona, and Serge Belongie. Visual recognition with humans in the loop. In *Computer Vision–ECCV 2010*, pages 438–451. Springer, 2010.

[16] Donald E Broadbent and Margaret HP Broadbent. From detection to identification: Response to multiple targets in rapid serial visual presentation. *Perception & psychophysics*, 42(2):105–113, 1987.

[17] Jerome Bruner. Culture and human development: A new look. *Human development*, 33(6):344–355, 1990.

[18] Razvan C Bunescu and Raymond J Mooney. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 724–731. Association for Computational Linguistics, 2005.

[19] Moira Burke and Robert Kraut. Using facebook after losing a job: Differential benefits of strong and weak ties. In *Proceedings of the 2013 conference on Computer supported cooperative work*, pages 1419–1430. ACM, 2013.

[20] Shu Cai and Kevin Knight. Smatch: An evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association of for Computational Linguistics*, 2013.

[21] Yang Cao, Changhu Wang, Zhiwei Li, Liqing Zhang, and Lei Zhang. Spatial-bag-of-features. In *CVPR '10. IEEE Conference on Computer Vision and Pattern Recognition, 2010.*, 2010.

[22] Stuart K Card, Allen Newell, and Thomas P Moran. The psychology of human-computer interaction. 1983.

[23] Angel X Chang, Manolis Savva, and Christopher D Manning. Learning spatial knowledge for text to 3D scene generation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, 2014.

[24] Angel X Chang, Manolis Savva, and Christopher D Manning. Semantic parsing for text to 3d scene generation. *ACL 2014*, page 17, 2014.

[25] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[26] Xinlei Chen, Ashish Shrivastava, and Arpan Gupta. Neil: Extracting visual knowledge from web data. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1409–1416. IEEE, 2013.

[27] Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. A unified model for word sense representation and disambiguation. In *EMNLP*, pages 1025–1035. Citeseer, 2014.

[28] Justin Cheng, Jaime Teevan, and Michael S Bernstein. Measuring crowdsourcing effort with error-time curves. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1365–1374. ACM, 2015.

[29] Lydia B Chilton, Greg Little, Darren Edge, Daniel S Weld, and James A Landay. Cascade: Crowdsourcing taxonomy creation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1999–2008. ACM, 2013.

[30] Myung Jin Choi, Joseph J Lim, Antonio Torralba, and Alan S Willsky. Exploiting hierarchical context on a large database of object categories. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 129–136. IEEE, 2010.

[31] Myung Jin Choi, Joseph J. Lim, Antonio Torralba, and Alan S. Willsky. Exploiting hierarchical context on a large database of object categories. In *IEEE Conference on Computer VIsion and Pattern Recognition (CVPR)*, 2010.

[32] Wongun Choi, Yu-Wei Chao, C. Pantofaru, and S. Savarese. Understanding indoor scenes using 3d geometric phrases. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, 2013.

[33] Wongun Choi, Yu-Wei Chao, Caroline Pantofaru, and Silvio Savarese. Understanding indoor scenes using 3d geometric phrases. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 33–40. IEEE, 2013.

[34] Grzegorz Chrupala, Akos Kadar, and Afra Alishahi. Learning language through pictures. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, 2015.

[35] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, 2007.

[36] Lacey Colligan, Henry WW Potts, Chelsea T Finn, and Robert A Sinkin. Cognitive workload changes for nurses transitioning from a legacy system with paper documentation to a commercial electronic health record. *International journal of medical informatics*, 84(7):469–476, 2015.

[37] Aron Culotta and Jeffrey Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Association for Computational Linguistics, 2004.

[38] Yann N Dauphin, Harm de Vries, Junyoung Chung, and Yoshua Bengio. Rmsprop and equilibrated adaptive learning rates for non-convex optimization. *arXiv preprint arXiv:1502.04390*, 2015.

[39] Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, 2014.

[40] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[41] Jia Deng, Olga Russakovsky, Jonathan Krause, Michael S Bernstein, Alex Berg, and Li Fei-Fei. Scalable multi-label annotation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3099–3102. ACM, 2014.

[42] Chaitanya Desai, Deva Ramanan, and Charless C Fowlkes. Discriminative models for multi-class object layout. *International Journal of Computer Vision*, 95(1):1–12, 2011.

[43] Santosh Divvala, Ali Farhadi, and Carlos Guestrin. Learning everything about anything: Webly-supervised visual concept learning. In *CVPR*, 2014.

[44] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(4):743–761, 2012.

[45] Desmond Elliott, Victor Lavrenko, and Frank Keller. Query-by-example image retrieval using visual dependency representations. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014.

[46] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[47] Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John Platt, et al. From captions to visual concepts and back. *arXiv preprint arXiv:1411.4952*, 2014.

[48] Ali Farhadi, Ian Endres, and Derek Hoiem. Attribute-centric recognition for cross-category generalization. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2352–2359. IEEE, 2010.

[49] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences from images. In *Computer Vision–ECCV 2010*, pages 15–29. Springer, 2010.

[50] Alireza Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1778–1785. IEEE, 2009.

[51] Ethan Fast, Daniel Steffee, Lucy Wang, Joel R Brandt, and Michael S Bernstein. Emergent, crowd-scale programming practice in the ide. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 2491–2500. ACM, 2014.

[52] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *CVPR-WS*, 2004.

[53] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[54] Li Fei-Fei, Asha Iyer, Christof Koch, and Pietro Perona. What do we perceive in a glance of a real-world scene? *Journal of vision*, 7(1):10, 2007.

[55] Vittorio Ferrari and Andrew Zisserman. Learning visual attributes. In *Advances in Neural Information Processing Systems*, pages 433–440, 2007.

[56] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79, 2010.

[57] Sanja Fidler and Aleš Leonardis. Towards scalable representations of object categories: Learning a hierarchy of parts. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[58] Chaz Firestone and Brian J Scholl. Cognition does not affect perception: Evaluating the evidence for top-down effects. *Behavioral and brain sciences*, pages 1–72, 2015.

[59] Matthew Fisher, Manolis Savva, and Pat Hanrahan. Characterizing structural relationships in scenes using graph kernels. In *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, pages 34:1–34:12. ACM, 2011.

[60] Matthew Fisher, Manolis Savva, and Pat Hanrahan. Characterizing structural relationships in scenes using graph kernels. In *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, 2011.

[61] Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014.

[62] Kenneth D Forbus. Qualitative process theory. *Artificial intelligence*, 24(1):85–168, 1984.

[63] David F Fouhey and C Lawrence Zitnick. Predicting object dynamics in scenes. CVPR, 2014.

[64] Carolina Galleguillos and Serge Belongie. Context based object categorization: A critical survey. *Computer Vision and Image Understanding*, 114(6):712–722, 2010.

[65] Carolina Galleguillos, Andrew Rabinovich, and Serge Belongie. Object categorization using co-occurrence, location and appearance. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[66] Haoyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu. Are you talking to a machine? dataset and methods for multilingual image question answering. *arXiv preprint arXiv:1505.05612*, 2015.

[67] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[68] Donald Geman, Stuart Geman, Neil Hallonquist, and Laurent Younes. Visual turing test for computer vision systems. *Proceedings of the National Academy of Sciences*, 112(12):3618–3623, 2015.

[69] Eric Gilbert and Karrie Karahalios. Predicting tie strength with social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 211–220. ACM, 2009.

[70] Ross Girshick. Fast r-cnn. *arXiv preprint arXiv:1504.08083*, 2015.

[71] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587. IEEE, 2014.

[72] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014.

[73] Christoph Goering, Erid Rodner, Alexander Freytag, and Joachim Denzler. Nonparametric part transfer for fine-grained recognition. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 2489–2496. IEEE, 2014.

[74] Stephen Gould, Jim Rodgers, David Cohen, Gal Elidan, and Daphne Koller. Multi-class segmentation with relative location prior. *International Journal of Computer Vision*, 80(3):300–316, 2008.

[75] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

[76] Sergio Guadarrama, Niveda Krishnamoorthy, Girish Malkarnenkar, Sarad Venugopalan, Randy Mooney, Trevor Darrell, and Kate Saenko. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2712–2719. IEEE, 2013.

[77] Zhou GuoDong, Su Jian, Zhang Jie, and Zhang Min. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 427–434. Association for Computational Linguistics, 2005.

[78] Abhinav Gupta and Larry S Davis. Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers. In *Computer Vision–ECCV 2008*, pages 16–29. Springer, 2008.

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

[80] Patrick J. Hayes. *The naive physics manifesto.* Institut pour les études sémantiques et cognitives/Université de Genève, 1978.

[81] Patrick J. Hayes. The second naive physics manifesto. *Theories of the Commonsense World*, pages 1–36, 1985.

[82] Marti A. Hearst, Susan T Dumais, Edgar Osman, John Platt, and Bernhard Scholkopf. Support vector machines. *Intelligent Systems and their Applications, IEEE*, 13(4):18–28, 1998.

[83] Jerry R Hobbs. Resolving pronoun references. *Lingua*, 44(4):311–338, 1978.

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

[85] Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *J. Artif. Int. Res.*, 47(1):853–899, May 2013.

[86] Derek Hoiem, Alexei A Efros, and Martial Hebert. Putting objects in perspective. *International Journal of Computer Vision*, 80(1):3–15, 2008.

[87] Chih-Sheng Johnson Hou, Natalya F Noy, and Mark A Musen. A template-based approach toward acquisition of logical sentences. In *Intelligent Information Processing*, pages 77–89. Springer, 2002.

[88] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on Faces in'Real-Life'Images: Detection, Alignment, and Recognition*, 2008.

[89] Marius Cătălin Iordan, Michelle R Greene, Diane M Beck, and Li Fei-Fei. Basic level category structure emerges gradually across human ventral visual cortex. *Journal of cognitive neuroscience*, 2015.

[90] Panagiotis G Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students*, 17(2):16–21, 2010.

[91] Panagiotis G Ipeirotis, Foster Provost, and Jing Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 64–67. ACM, 2010.

[92] Lilly C Irani and M Silberman. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 611–620. ACM, 2013.

[93] Phillip Isola, Joseph J Lim, and Edward H Adelson. Discovering states and transformations in image collections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1383–1391, 2015.

[94] Hamid Izadinia, Fereshteh Sadeghi, and Alireza Farhadi. Incorporating scene context and object layout into appearance modeling. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 232–239. IEEE, 2014.

[95] Suyog Dutt Jain and Kristen Grauman. Predicting sufficient annotation strength for interactive foreground segmentation. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1313–1320. IEEE, 2013.

[96] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2011.

[97] H. Jegou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2012.

[98] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[99] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3668–3678, 2015.

[100] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[101] Tatiana Josephy, Matt Lease, and Praveen Paritosh. Crowdscale 2013: Crowdsourcing at scale workshop report. 2013.

[102] Dan Jurafsky and James H Martin. *Speech & language processing*. Pearson Education India, 2000.

[103] Ece Kamar, Severin Hacker, and Eric Horvitz. Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 467–474. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[104] David R Karger, Sewoong Oh, and Devavrat Shah. Budget-optimal crowdsourcing using low-rank matrix approximations. In *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*, pages 284–291. IEEE, 2011.

[105] David R Karger, Sewoong Oh, and Devavrat Shah. Budget-optimal task allocation for reliable crowdsourcing systems. *Operations Research*, 62(1):1–24, 2014.

[106] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. *arXiv preprint arXiv:1412.2306*, 2014.

[107] Andrej Karpathy, Armand Joulin, and Li Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. *CoRR*, abs/1406.5679, 2014.

[108] Ryan Kiros, Ruslan Salakhutdinov, and Rich Zemel. Multimodal neural language models. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 595–603, 2014.

[109] Ryan Kiros, Ruslan Salakhutdinov, and Richard Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *Transactions of the Association for Computational Linguistics*, 2015.

[110] Aniket Kittur, Ed H Chi, and Bongwon Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–456. ACM, 2008.

[111] Dan Klein and Christopher D Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003.

[112] Philipp Krähenbühl and Vladlen Koltun. Geodesic object proposals. In *Computer Vision–ECCV 2014*, pages 725–739. Springer, 2014.

[113] Ranjay Krishna, Kenji Hata, Stephanie Chen, Joshua Kravitz, David A Shamma, Li Fei-Fei, and Michael S. Bernstein. Embracing error to enable rapid crowdsourcing. In *CHI'16-SIGCHI Conference on Human Factors in Computing System*, 2016.

[114] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalanditis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. 2016.

[115] Jayant Krishnamurthy and Thomas Kollar. Jointly learning to parse and perceive: Connecting natural language to the physical world. *TACL*, 1:193–206, 2013.

[116] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[117] Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. Baby talk: Understanding and generating image descriptions. In *Proceedings of the 24th CVPR*. Citeseer, 2011.

[118] M Pawan Kumar and Daphne Koller. Efficiently selecting regions for scene understanding. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3217–3224. IEEE, 2010.

[119] Ranjitha Kumar, Arvind Satyanarayan, Cesar Torres, Maxine Lim, Salman Ahmad, Scott R Klemmer, and Jerry O Talton. Webzeitgeist: design mining the web. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3083–3092. ACM, 2013.

[120] Lubor Ladicky, Chris Russell, Pushmeet Kohli, and Philip HS Torr. Graph cut based inference with co-occurrence statistics. In *Computer Vision–ECCV 2010*, pages 239–253. Springer, 2010.

[121] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, 2001.

[122] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, 2001.

[123] C Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot learning of object categories. *PAMI*, 2013.

[124] Christoph H. Lampert. Detecting objects in large image collections and videos by efficient subimage retrieval. In *Computer Vision, 2009 IEEE 12th International Conference on*, 2009.

[125] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 951–958. IEEE, 2009.

[126] Gierad Laput, Walter S Lasecki, Jason Wiese, Robert Xiao, Jeffrey P Bigham, and Chris Harrison. Zensors: Adaptive, rapidly deployable, human-intelligent sensor feeds. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1935–1944. ACM, 2015.

[127] Walter Lasecki, Christopher Miller, Adam Sadilek, Andrew Abumoussa, Donato Borrello, Raja Kushalnagar, and Jeffrey Bigham. Real-time captioning by groups of non-experts. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pages 23–34. ACM, 2012.

[128] Walter S Lasecki, Kyle I Murray, Samuel White, Robert C Miller, and Jeffrey P Bigham. Real-time crowd control of existing interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 23–32. ACM, 2011.

[129] Claudia Leacock, George A Miller, and Martin Chodorow. Using corpus statistics and wordnet relations for sense identification. *Computational Linguistics*, 24(1):147–165, 1998.

[130] Rémi Lebret, Pedro O Pinheiro, and Ronan Collobert. Phrase-based image captioning. *arXiv preprint arXiv:1502.03671*, 2015.

[131] Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916, 2013.

[132] Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 28–34. Association for Computational Linguistics, 2011.

[133] David D. Lewis and Philip J. Hayes. Guest editorial. *ACM Transactions on Information Systems*, 12(3):231, July 1994.

[134] Fei Fei Li, Rufin VanRullen, Christof Koch, and Pietro Perona. Rapid natural scene categorization in the near absence of attention. *Proceedings of the National Academy of Sciences*, 99(14):9596–9601, 2002.

[135] Li-Jia Li, Richard Socher, and Li Fei-Fei. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *CVPR*, 2009.

[136] Tao Li and Mitsunori Ogihara. Detecting emotion in music. In *ISMIR*, volume 3, pages 239–240, 2003.

[137] Liang Liang and Kristen Grauman. Beyond comparing image pairs: Setwise active learning for relative attributes. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 208–215. IEEE, 2014.

[138] Dahua Lin, Sanja Fidler, Chen Kong, and Raquel Urtasun. Visual semantic search: Retrieving videos via complex textual queries. In *CVPR*, 2014.

[139] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014*, pages 740–755. Springer, 2014.

[140] Ying Liu, Dengsheng Zhang, Guojun Lu, and Wei-Ying Ma. A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40(1):262–282, 2007.

[141] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[142] Cewu Lu, Ranjay Krishna, Michael S. Bernstein, and Li Fei-Fei. Visual relationship detection using language priors. 2016.

[143] Lin Ma, Zhengdong Lu, and Hang Li. Learning to answer questions from image using convolutional neural network. *arXiv preprint arXiv:1506.00333*, 2015.

[144] Subhransu Maji, Lubomir Bourdev, and Jitendra Malik. Action recognition from a distributed representation of pose and appearance. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3177–3184. IEEE, 2011.

[145] Mateusz Malinowski and Mario Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in Neural Information Processing Systems*, pages 1682–1690, 2014.

[146] Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. Ask your neurons: A neural-based approach to answering questions about images. *arXiv preprint arXiv:1505.01121*, 2015.

[147] Tomasz Malisiewicz, Alexei Efros, et al. Recognition by association via learning per-exemplar distances. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[148] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.

[149] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Heng Huangzhi, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-RNN). In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*, 2015.

[150] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Alan L Yuille. Explain images with multimodal recurrent neural networks. *arXiv preprint arXiv:1410.1090*, 2014.

[151] Adam Marcus and Aditya Parameswaran. *Crowdsourced data management: industry and academic perspectives*. Foundations and Trends in Databases, 2015.

[152] Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

[153] David Martin, Benjamin V Hanrahan, Jacki O'Neill, and Neha Gupta. Being a turker. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, pages 224–235. ACM, 2014.

[154] Winter Mason and Siddharth Suri. Conducting behavioral research on amazons mechanical turk. *Behavior research methods*, 44(1):1–23, 2012.

[155] Thomas Mensink, Efstratios Gavves, and Cees GM Snoek. Costa: Co-occurrence statistics for zero-shot classification. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 2441–2448. IEEE, 2014.

[156] Thomas Mensink, Efstratios Gavves, and Cees G.M. Snoek. Costa: Co-occurrence statistics for zero-shot classification. In *CVPR*, 2014.

[157] Rada Mihalcea, Timothy Anatolievich Chklovski, and Adam Kilgarriff. The senseval-3 english lexical sample task. Association for Computational Linguistics, 2004.

[158] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[159] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[160] George A Miller and Walter G Charles. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28, 1991.

[161] Margaret Mitchell, Xufeng Han, Jesse Dodge, Alyssa Mensch, Amit Goyal, Alex Berg, Kota Yamaguchi, Tamara Berg, Karl Stratos, and Hal Daumé III. Midge: Generating image descriptions from computer vision detections. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 747–756. Association for Computational Linguistics, 2012.

[162] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[163] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[164] Feng Niu, Ce Zhang, Christopher Ré, and Jude Shavlik. Elementary: Large-scale knowledge-base construction via machine learning and statistical inference. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(3):42–73, 2012.

[165] Feng Niu, Ce Zhang, Christopher Ré, and Jude W Shavlik. Deepdive: Web-scale knowledge-base construction using statistical learning and inference. In *VLDS*, pages 25–28, 2012.

[166] Rick Nouwen. Plurality. In Paul Dekker and Maria Aloni, editors, *Cambridge Handbook of Semantics*. Cambridge University Press, Cambridge, 2015.

[167] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001.

[168] Vicente Ordonez, Girish Kulkarni, and Tamara L. Berg. Im2text: Describing images using 1 million captioned photographs. In J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1143–1151. Curran Associates, Inc., 2011.

[169] Alok Ranjan Pal and Diganta Saha. Word sense disambiguation: a survey. *arXiv preprint arXiv:1508.01346*, 2015.

[170] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135, 2008.

[171] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[172] Devi Parikh and Kristen Grauman. Relative attributes. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 503–510. IEEE, 2011.

[173] Amar Parkash and Devi Parikh. Attributes for classifier feedback. In *Computer Vision–ECCV 2012*, pages 354–368. Springer, 2012.

[174] Genevieve Patterson, Chen Xu, Hang Su, and James Hays. The sun attribute database: Beyond categories for deeper scene understanding. *International Journal of Computer Vision*, 108(1-2):59–81, 2014.

[175] Mausam Daniel Peng Dai and S Weld. Decision-theoretic control of crowd-sourced workflows. In *In the 24th AAAI Conference on Artificial Intelligence (AAAI10*. Citeseer, 2010.

[176] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 2014.

[177] F. Perronnin, Y. Liu, J. Sanchez, and H. Poirier. Large-scale image retrieval with compressed fisher vectors. In *CVPR*, 2010.

[178] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *Computer Vision–ECCV 2010*, pages 143–156. Springer, 2010.

[179] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in large margin classifiers*, 1999.

[180] Bryan Plummer, Liwei Wang, Chris Cervantes, Juan Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k Entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. *arXiv preprint arXiv:1505.04870*, 2015.

[181] Mary C Potter. Short-term conceptual memory for pictures. *Journal of experimental psychology: human learning and memory*, 2(5):509, 1976.

[182] Mary C Potter and Ellen I Levy. Recognition memory for a rapid sequence of pictures. *Journal of experimental psychology*, 81(1):10, 1969.

[183] Alessandro Prest, Cordelia Schmid, and Vittorio Ferrari. Weakly supervised learning of interactions between humans and objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(3):601–614, 2012.

[184] Danfeng Qin, Christian Wengert, and Luc Van Gool. Query adaptive similarity for large scale object retrieval. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[185] Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. Objects in context. In *Computer vision, 2007. ICCV 2007. IEEE 11th international conference on*, pages 1–8. IEEE, 2007.

[186] Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 492–501. Association for Computational Linguistics, 2010.

[187] Vignesh Ramanathan, Congcong Li, Jia Deng, Wei Han, Zhen Li, Kunlong Gu, Yang Song, Samy Bengio, Chuck Rossenberg, and Li Fei-Fei. Learning semantic relationships for better action retrieval in images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1100–1109, 2015.

[188] Adam Reeves and George Sperling. Attention gating in short-term visual memory. *Psychological review*, 93(2):180, 1986.

[189] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics*, 1:25–36, 2013.

[190] Mengye Ren, Ryan Kiros, and Richard Zemel. Image question answering: A visual semantic embedding model and a new dataset. *arXiv preprint arXiv:1505.02074*, 2015.

[191] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

[192] Marcus Rohrbach, Wei Qiu, Igor Titov, Stefan Thater, Manfred Pinkal, and Bernt Schiele. Translating video content to natural language descriptions. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 433–440. IEEE, 2013.

[193] Sascha Rothe and Hinrich Schütze. Autoextend: Extending word embeddings to embeddings for synsets and lexemes. *arXiv preprint arXiv:1507.01127*, 2015.

[194] M. Ruggero Ronchi and P. Perona. Describing Common Human Visual Actions in Images. *ArXiv e-prints*, June 2015.

[195] Yong Rui, Thomas S Huang, and Shih-Fu Chang. Image retrieval: Current techniques, promising directions, and open issues. *Journal of visual communication and image representation*, 10(1):39–62, 1999.

[196] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2014.

[197] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, pages 1–42, April 2015.

[198] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, pages 1–42, 2014.

[199] Olga Russakovsky, Li-Jia Li, and Li Fei-Fei. Best of both worlds: human-machine collaboration for object annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2121–2131, 2015.

[200] Bryan C Russell, William T Freeman, Alexei Efros, Josef Sivic, Andrew Zisserman, et al. Using multiple segmentations to discover objects and their extent in image collections. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1605–1614. IEEE, 2006.

[201] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.

[202] Fereshteh Sadeghi, Santosh K Divvala, and Ali Farhadi. Viske: Visual knowledge extraction and question answering by visual verification of relation phrases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1456–1464, 2015.

[203] Mohammad Amin Sadeghi and Ali Farhadi. Recognition using visual phrases. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1745–1752. IEEE, 2011.

[204] Ruslan Salakhutdinov, Antonio Torralba, and Josh Tenenbaum. Learning to share visual appearance for multiclass object detection. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1481–1488. IEEE, 2011.

[205] Niloufar Salehi, Lilly C Irani, and Michael S Bernstein. We are dynamo: Overcoming stalling and friction in collective action for crowd workers. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1621–1630. ACM, 2015.

[206] Roger C Schank and Robert P Abelson. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures.* Psychology Press, 2013.

[207] Robert E Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2):135–168, 2000.

[208] Karin Kipper Schuler. *Verbnet: A Broad-coverage, Comprehensive Verb Lexicon.* PhD thesis, Philadelphia, PA, USA, 2005. AAI3179808.

[209] Sebastian Schuster, Ranjay Krishna, Angel Chang, Li Fei-Fei, and Christopher D Manning. Generating semantically precise scene graphs from textual descriptions for improved image retrieval. 2015.

[210] Prem Seetharaman and Bryan Pardo. Crowdsourcing a reverberation descriptor map. In *Proceedings of the ACM International Conference on Multimedia*, pages 587–596. ACM, 2014.

[211] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[212] Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622. ACM, 2008.

[213] Aashish Sheshadri and Matthew Lease. Square: A benchmark for research on computing crowd consensus. In *First AAAI Conference on Human Computation and Crowdsourcing*, 2013.

[214] Behjat Siddiquie, Rogerio S Feris, and Larry S Davis. Image ranking and retrieval based on multi-attribute queries. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[215] Raffaele Simone and Francesca Masini. On light nouns. *Word Classes: Nature, typology and representations*, 332:51, 2014.

[216] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[217] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[218] Josef Sivic, Bryan C Russell, Alexei Efros, Andrew Zisserman, William T Freeman, et al. Discovering objects and their location in images. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 370–377. IEEE, 2005.

[219] Padhraic Smyth, Michael C Burl, Usama M Fayyad, and Pietro Perona. Knowledge discovery in large image databases: Dealing with uncertainties in ground truth. In *KDD Workshop*, pages 109–120, 1994.

[220] Padhraic Smyth, Usama Fayyad, Michael Burl, Pietro Perona, and Pierre Baldi. Inferring ground truth from subjective labelling of venus images. 1995.

[221] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 254–263. Association for Computational Linguistics, 2008.

[222] Richard Socher and Li Fei-Fei. Connecting modalities: Semi-supervised segmentation and annotation of images using unaligned text corpora. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 966–973. IEEE, 2010.

[223] Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211. Association for Computational Linguistics, 2012.

[224] Richard Socher, Andrej Karpathy, Quoc V Le, Christopher D Manning, and Andrew Y Ng. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2, 2014.

[225] Richard Socher, Q Le, C Manning, and A Ng. Grounded compositional semantics for finding and describing images with sentences. In *NIPS Deep Learning Workshop*, 2013.

[226] Zheng Song, Qiang Chen, Zhongyang Huang, Yang Hua, and Shuicheng Yan. Contextualizing object detection and classification. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1585–1592. IEEE, 2011.

[227] HCI Stanford. Daemo: a self-governed crowdsourcing marketplace.

[228] Henrik Stewenius and Steinar H. Gunderson Julien Pilet. Size matters: Exhaustive geometric verification for image retrieval. In *ECCV*.

[229] Hao Su, Jia Deng, and Li Fei-Fei. Crowdsourcing annotations for visual object detection. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[230] Hao Su, Jia Deng, and Li Fei-Fei. Crowdsourcing annotations for visual object detection. In *AAAI Technical Report, 4th Human Computation Workshop*, 2012.

[231] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

[232] Omer Tamuz, Ce Liu, Serge Belongie, Ohad Shamir, and Adam Tauman Kalai. Adaptively learning the crowd kernel. *arXiv preprint arXiv:1105.1033*, 2011.

[233] Jesse Thomason, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Raymond Mooney. Integrating language and vision to generate natural language descriptions of videos in the wild. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING), August*, 2014.

[234] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. The new data and new challenges in multimedia research. *arXiv preprint arXiv:1503.01817*, 2015.

[235] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Commun. ACM*, 59(2):64–73, January 2016.

[236] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2), 2016. To Appear.

[237] Antonio Torralba, Rob Fergus, and William T Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(11):1958–1970, 2008.

[238] Lorenzo Torresani, Martin Szummer, and Andrew Fitzgibbon. Efficient object category recognition using classemes. In *European Conference on Computer Vision (ECCV)*, pages 776–789, September 2010.

[239] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[240] Manik Varma and Andrew Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1-2):61–81, 2005.

[241] Sudheendra Vijayanarasimhan, Prateek Jain, and Kristen Grauman. Far-sighted active learning on a budget for image and video recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3035–3042. IEEE, 2010.

[242] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *arXiv preprint arXiv:1411.4555*, 2014.

[243] Carl Vondrick, Donald Patterson, and Deva Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, 101(1):184–204, 2013.

[244] Catherine Wah, Steve Branson, Pietro Perona, and Serge Belongie. Multiclass recognition and part localization with humans in the loop. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2524–2531. IEEE, 2011.

[245] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[246] Catherine Wah, Grant Van Horn, Steve Branson, Subhrajyoti Maji, Pietro Perona, and Serge Belongie. Similarity comparisons for interactive fine-grained categorization. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 859–866. IEEE, 2014.

[247] Chuan Wang, Nianwen Xue, Sameer Pradhan, and Sameer Pradhan. A transition-based algorithm for amr parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics Human Language Technologies (NAACL HLT 2015)*, 2015.

[248] Erich Weichselgartner and George Sperling. Dynamics of automatic and controlled visual attention. *Science*, 238(4828):778–780, 1987.

[249] Peter Welinder, Steve Branson, Pietro Perona, and Serge J Belongie. The multidimensional wisdom of crowds. In *Advances in neural information processing systems*, pages 2424–2432, 2010.

[250] Keenon Werling, Gabor Angeli, and Christopher D. Manning. Robust subgraph generation improves abstract meaning representation parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, 2015.

[251] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, pages 2035–2043, 2009.

[252] Jacob O Wobbrock, Jodi Forlizzi, Scott E Hudson, and Brad A Myers. Webthumb: interaction techniques for small-screen browsers. In *Proceedings of the 15th annual ACM symposium on User interface software and technology*, pages 205–208. ACM, 2002.

[253] Jianxiong Xiao, James Hays, Krista Ehinger, Aude Oliva, Antonio Torralba, et al. Sun database: Large-scale scene recognition from abbey to zoo. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 3485–3492. IEEE, 2010.

[254] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *CoRR*, abs/1502.03044, 2015.

[255] Jimei Yang, Brian Price, Scott Cohen, and Ming-Hsuan Yang. Context driven scene parsing with attention to rare classes. In *CVPR*, 2014.

[256] Bangpeng Yao and Li 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.

[257] Bangpeng Yao and Li 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.

[258] Bangpeng Yao, Aditya Khosla, and Li Fei-Fei. Classifying actions and measuring action similarity by modeling the mutual context of objects and human poses. *a) A*, 1(D2):D3, 2011.

[259] Benjamin Yao, Xiong Yang, and Song-Chun Zhu. Introduction to a large-scale general purpose ground truth database: methodology, annotation tool and benchmarks. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 169–183. Springer, 2007.

[260] Jian Yao, Sanja Fidler, and Raquel Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 702–709. IEEE, 2012.

[261] Z.Y. Yao, X. Yang, and S.C. Zhu. Introduction to a large scale general purpose groundtruth dataset: methodology, annotation tool, and benchmarks. In *EMMCVPR*, 2007.

[262] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

[263] Licheng Yu, Eunbyung Park, Alexander C. Berg, and Tamara L. Berg. Visual Madlibs: Fill in the blank Image Generation and Question Answering. *arXiv preprint arXiv:1506.00278*, 2015.

[264] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. Relation classification via convolutional deep neural network. In *Proceedings of COLING*, pages 2335–2344, 2014.

[265] Xiao Zhang, Zhiwei Li, Lei Zhang, Wei-Ying Ma, and Heung-Yeung Shum. Efficient indexing for large scale visual search. In *Computer Vision, 2009 IEEE 12th International Conference on*, 2009.

[266] Yibiao Zhao and Song-Chun Zhu. Scene parsing by integrating function, geometry and appearance models. In *CVPR*, 2013.

[267] Dengyong Zhou, Sumit Basu, Yi Mao, and John C Platt. Learning from the wisdom of crowds by minimax entropy. In *Advances in Neural Information Processing Systems*, pages 2195–2203, 2012.

[268] GuoDong Zhou, Min Zhang, Dong Hong Ji, and Qiaoming Zhu. Tree kernel-based relation extraction with context-sensitive structured parse tree information. *EMNLP-CoNLL 2007*, page 728, 2007.

[269] Jun Zhu, Zaiqing Nie, Xiaojiang Liu, Bo Zhang, and Ji-Rong Wen. Statsnowball: a statistical approach to extracting entity relationships. In *Proceedings of the 18th international conference on World wide web*, pages 101–110. ACM, 2009.

[270] Yuke Zhu, Alireza Fathi, and Li Fei-Fei. Reasoning about Object Affordances in a Knowledge Base Representation. In *European Conference on Computer Vision*, 2014.

[271] Yuke Zhu, Ce Zhang, Christopher Ré, and Li Fei-Fei. Building a Large-scale Multimodal Knowledge Base System for Answering Visual Queries. In *arXiv preprint arXiv:1507.05670*, 2015.

[272] C Lawrence Zitnick and Devi Parikh. Bringing semantics into focus using visual abstraction. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3009–3016. IEEE, 2013.

[273] C Lawrence Zitnick, Devi Parikh, and Lucy Vanderwende. Learning the visual interpretation of sentences. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1681–1688. IEEE, 2013.