TEXTUAL ENTAILMENT FROM IMAGE CAPTION DENOTATIONS

BY

ALICE YINGMING LAI

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Doctoral Committee:

Associate Professor Julia Hockenmaier, Chair
Professor Katrin Erk, University of Texas at Austin
Professor Dan Roth
Professor ChengXiang Zhai

ABSTRACT

Understanding the meaning of linguistic expressions is a fundamental task of natural language processing. While distributed representations have become a powerful technique for modeling lexical semantics, they have traditionally relied on ungrounded text corpora to identify semantically similar words. In contrast, this thesis explicitly models the denotation of linguistic expressions by building representations from grounded image captions. This allows us to use descriptions of the world to learn connections that would be difficult to identify in text-based corpora. In particular, we explore novel approaches to entailment that capture everyday world knowledge missing from other NLP tasks, on both existing datasets and our own new dataset. We also present a novel embedding model that produces phrase representations that are informed by our grounded representation. We conclude with an analysis of how grounded embeddings differ from standard distributional embeddings and suggestions for future refinement of this approach.

ACKNOWLEDGMENTS

When I first arrived at Illinois, Julia Hockenmaier welcomed me into her lab and gave me hours of her time even before I decided to work with her. The work in this thesis was all inspired, directly or indirectly, by papers I read and conversations we had that first year as she encouraged me to find a topic I was passionate about. Julia's patience and attention to detail have shaped my work in countless ways, and my time at Illinois would not have been the same without her.

I would also like to thank my committee for their time and effort. Katrin Erk has been thoughtful and kind in her comments, and encouraged me to consider broader research questions. Cheng Zhai is always enthusiastic in his feedback, and reminded me that there are applications beyond NLP. Dan Roth has been very generous with his time since the first class I took from him six years ago, and I truly appreciate the insights and advice he has given me.

I would not have made it through this degree without the support of the friends I made at Illinois. Jason Rock has been a great source of entertainment and an even better friend since our first day of class together. Daphne Tsatsoulis, my clothing twin, always reminded me how important it is to take breaks. Kristen Vaccaro convinced me to get a cat and probably saved my sanity. Pooya Khorrami reminded me to face my weaknesses. Chris Cervantes was my podcast roadtrip buddy. Christos Christodoulopoulos was always ready with tea and conversation to offset cold winter evenings. Mallory Casperson and Brett Jones introduced me to dogs and projectors.

It is important to have good labmates to share the stresses of graduate school. I am lucky to have spent some of that time with Ryan Musa, Micah Hodosh, Peter Young, Anjali Narayan-Chen, and Prashant Jayannavar. I am glad to have gone through classes and quals and beyond with Stephen Mayhew, Joe Degol, Daniel Khashabi, Subhro Roy, and Chen-Tse Tsai. Thanks as well to my vision buddies – including Tanmay Gupta, Aditya Deshpande, Kevin Shih, and Raj Kataria – who helped me out with deep learning and kept me up-to-date with the developments in other applied AI fields.

I discovered GradSWE at Illinois at a time when I was struggling to make progress and stay on track. I met a lot of amazing people there who reminded me that there is life, both personal and professional, outside of a computer lab.

I also want to thank Joel Tetreault, who was an amazing mentor and collaborator during my time at Grammarly, and who reminded me to have confidence in my ideas and my work.

I owe a great deal to my parents both for their unwavering support and for their practicality, two things that sometimes seem at odds when pursuing a Ph.D., but they made it work. And to the rest of my family – particularly Katie Bent, Sheila Houlahan, and Shirley Chen – thanks for always being eager to pick up where we left off.

Finally, to Yonatan Bisk: none of this would have been possible without you.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

The ability to draw inferences is an important part of language understanding. When we have a conversation with someone, for example, or when we read a newspaper, we use our existing knowledge of the world to infer a great deal of information that was unstated by the speaker or the text. This process allows us to have everyday conversations or write an article without first listing all the facts and assumptions that we hold to be true. For descriptive language, these types of inferences mean that even simple descriptions can evoke rich, complex mental imagery. A description of *people shopping in a supermarket* comes with the expectation that these people are probably pushing shopping carts and surrounded by aisles of shelved food.

In natural language processing, we have established textual entailment as the task that evaluates the ability of our models to infer unstated information from text. A model that can accurately judge whether one text entails another should also be successful at other tasks that we are interested in, such as question answering or summarization. This line of research has resulted in models that use a broad range of approaches to tackle inference, from alignment and rewriting rule extraction to lexical similarity and more recent end-to-end neural network models.

In this thesis, we focus on applying denotational similarity, an image-grounded similarity metric, to textual entailment. We first introduced denotational similarity to facilitate natural language inference about everyday scenes. Compared to standard distributional similarity that captures whether words occur in similar linguistic contexts, denotational similarity is grounded in common images and expresses whether words can be used to describe the same event. We expand on that initial work to demonstrate the usefulness of denotational similarity to textual entailment, both as an explicit feature and in the form of novel phrase embeddings.

1.1 THESIS STATEMENT

In this thesis, we explore textual entailment from the perspective of image-grounded denotation, an effective representation for tasks that require natural language representation. We introduce an embedding representation that expresses the denotational similarity of everyday actions as vectors in an interpretable embedding space, and we present the first model that automatically produces these vectors for unseen phrases.

We also present a new textual entailment corpus where the premise is comprised of sen-

tences that all share a common grounded image denotation. Our aim is to evaluate a model's ability to build a unified representation of the denotation of a set of sentences. Our denotational embedding model captures semantic relationships that are fundamentally different from standard distributional embeddings and are demonstrably effective for natural language inference.

1.2 CONTRIBUTIONS

Feature-Based Application of Denotational Similarity In Chapter 3 and Chapter 4, we present models that use denotational similarity features as part of a system for textual entailment and semantic similarity. We demonstrate that denotational similarity features over constituent phrase pairs contribute to state-of-the-art performance. Furthermore, we present ablation results that show that denotational similarity is complementary to distributional similarity features on these tasks.

Denotational Embedding Models In Chapter 5, we propose a framework to map denotational probabilities to a structured vector embedding space that captures denotational set relationships. We present the first model that can produce these denotational embeddings for new phrases, which allows us to predict the denotational conditional probability of a hypothesis sentence given a premise sentence for a textual entailment task. In Chapter 7, we introduce another denotational embedding model that outperforms neural sentence encoding models for entailment without the need for a constituent extraction step.

Multiple Premise Entailment Task To explore the space of image caption denotation, in Chapter 6 we present a new textual entailment dataset where the premise text is a set of captions that describe the same image. This requires that models aggregate information from multiple independently written captions in order to predict the entailment relationship between premise and hypothesis.

Analysis of Denotational Embeddings In Chapter 7, we analyze our denotational embeddings to account for the types of semantic relationships that they capture. We confirm that denotational embeddings, like the similarities computed from the denotation graph, capture entailment-like relationships within scenes. This distinguishes denotational embeddings from distributional embeddings, which tend to capture synonym relationships. We also show that denotational embeddings capture visual scene information that is not available to standard text-based models.

Our goal in this thesis is to continue the development of denotational similarity through new models and datasets that allow us to apply it to a broader range of scenarios. We show that we can train neural embedding models to produce denotational representations for new phrases. The resulting phrase embeddings are another step towards using denotation for more complicated inference tasks in multiple domains. They emphasize semantic relationships that are complementary to those captured by standard distributional representations, creating opportunities to build future semantic representations that combine both types of information.

CHAPTER 2: TEXTUAL ENTAILMENT

Inference, the process by which we reach a new conclusion from the provided evidence, is an important part of human reasoning and natural language. Inference is how we determine that a new statement is true based on previous information. In Natural Language Processing, we have recast this process as the *textual entailment* task, in which the goal is to conclude whether one text is true or false based on the information given in a prior text. Textual entailment has become an important semantic evaluation, first proposed by Dagan and Glickman [1], popularized initially by the Recognizing Textual Entailment challenges [2] and more recently by the much larger datasets SICK [3] and SNLI [4]. In this chapter, we present some background information on textual entailment datasets and models. The contributions of this thesis begin in Chapter 3.

Generally, textual entailment consists of two texts: a premise text p and a hypothesis text h. The premise p entails the hypothesis h if a human who read p would assume that h is probably true. This definition differs from the logic-based definition of entailment from the field of linguistics. A definition of entailment based on formal semantics states that the premise p entails another text h if h is true in every single possible situation where p is true. This is true for some examples of textual entailment, such as:

PREMISE: A man is doing a trick on a skateboard.
HYPOTHESIS: A person is doing a trick on a skateboard.

$\Rightarrow$ ENTAILMENT

because all *men* are also *people* and therefore the hypothesis is always true when the premise is true. However, for other examples in standard textual entailment datasets, this is not the case:

PREMISE: A senior is waiting at the window of a restaurant that serves sandwiches.
HYPOTHESIS: A person waits to be served his food.

$\Rightarrow$ ENTAILMENT

In the above example, there exist other conceivable scenarios where the person is waiting for his change after paying for the food, or is waiting at the window for some reason other than being a customer of the restaurant. However, most people who read the premise sentence assume that a person waiting at a restaurant window is indeed waiting for food, and label the sentence pair as ENTAILMENT. In general, we are interested in this definition of entailment: what people assume to be true given the information they currently have. We want NLP tools to model human understanding of and assumptions about language. This

approach to textual entailment has the potential to be more useful to downstream tasks, in contrast to a strict logical definition of entailment that does not correspond to how people use language.

Dagan et al. [5] describe textual entailment as covering a broad definition of inference, that can be classified into several types. One type of entailment involves deriving *new* information from the premise by reasoning about what the premise implies. For example, *"Barack Obama was the 44th President of the United States"* entails *"Barack Obama is a U.S. citizen."* The premise does not state outright that Obama is a citizen, but we can infer this information based on the premise and what we know to be true about the world (that *presidents are citizens of their countries*).

Other cases of textual entailment, however, do not involve deriving new information from the premise. Instead, the hypothesis may state a generalization of the information in the premise, or may be synonymous to the premise. For example, *drugs are used to slow the progress of Alzheimer's* entails *drugs are used to treat Alzheimer's. Treating a disease* is a more general expression that does not state whether the disease is slowed, halted, or cured thanks to the drugs. A hypothesis that paraphrases the premise may involve replacing a word with a synonym, or could involve more complex paraphrasing. This type of inference involves capturing the *variability* of language: how the same information may be stated in many different ways and with different levels of specificity. This variability of language is what we aim to capture in creating the FLICKR30K dataset and the denotation graph, as we will describe in Chapter 3.

Textual entailment was conceived as a unified framework for semantic inference that could be extended to downstream NLP tasks. For example, in a reading comprehension scenario, the system should select a response that is entailed by the provided text. A summarization system should not generate a summary that is not entailed by the text it is supposed to summarize, so a textual entailment model could provide a very useful check on valid outputs.

In this chapter, we survey existing textual entailment datasets, several of which we will use for evaluation in this thesis. We also present a brief overview of approaches to these datasets.

2.1 CORPORA

2.1.1 Recognizing Textual Entailment

Initial work on textual entailment was driven by the Recognizing Textual Entailment (RTE) datasets [2], a series of challenges presenting textual entailment data for a range of

Premise	Hypothesis	Label
Drew Walker, NHS Tayside's public health director, said: "It is important to stress that this is not a confirmed case of rabies."	A case of rabies was confirmed.	NON-ENTAILMENT
About two weeks before the trial started, I was in Shapiro's office in Century City.	Shapiro works in Century City.	ENTAILMENT
The drugs that slow down or halt Alzheimer's disease work best the earlier you administer them.	Alzheimer's disease is treated using drugs.	ENTAILMENT
Arabic, for example, is used densely across North Africa and from the Eastern Mediterranean to the Philippines, as the key language of the Arab world and the primary vehicle of Islam.	Arabic is the primary language of the Philippines.	NON-ENTAILMENT

Table 2.1: Examples of development data from RTE-2.

domains and potential applications. Some tasks involve validating the output of question answering or text summarization systems, areas where textual entailment models could prove useful. In total, there are about seven thousand entailment examples across the RTE datasets [6].

RTE covers a broad range of inference types: lexical substitution, coreference, named entity recognition, logical reasoning, the ability to compare numbers, and more general world knowledge (e.g. knowing that *"people work in their offices"* and *"presidents are citizens of their countries"*) [5]. These problems can be very complex and may assume a high degree of world knowledge, e.g. knowing that a company that has *"filed for its IPO"* has *"gone public."* As such, models may require named entity recognition, syntactic parsing, semantic role labeling, coreference resolution, and other components of the full NLP pipeline.

The first RTE challenges presented textual entailment as a binary task: the premise either *does* or *does not* entail the hypothesis. Table 2.1 contains a few of these binary entailment examples. Later challenges increased the number of classes to make the distinction between a hypothesis that *contradicts* the premise and one that has a *neutral* or independent relationship to the premise. This three-way classification task was first introduced in RTE-4 [7]. Many recent textual entailment datasets have also treated textual entailment as a three-class problem, including the two datasets that we will discuss in the next sections.

Premise	Hypothesis	Label
A boy is standing in the cold water.	A boy is standing in the water.	ENTAILMENT
The boy is sitting near the blue ocean.	The boy is wading through the blue ocean.	CONTRADICTION
A boy is standing in the water.	The kid is swimming through the blue ocean.	NEUTRAL

Table 2.2: Examples of training data from SICK.

Premise	Hypothesis	Label
An older man is drinking orange juice at a restaurant.	A man is drinking juice.	ENTAILMENT
An older man is drinking orange juice at a restaurant.	A man in a restaurant is waiting for his meal to arrive.	NEUTRAL
An older man is drinking orange juice at a restaurant.	Two women are at a restaurant drinking wine.	CONTRADICTION

Table 2.3: Examples of training data from SNLI.

2.1.2 Sentences Involving Compositional Knowledge

The Sentences Involving Compositional Knowledge (SICK) dataset [3, 8] was created as a simplified textual entailment task that does not require as much world knowledge as RTE and relies less on other NLP tasks like coreference resolution and named entity recognition. A few examples are presented in Table 2.2.

SICK contains almost ten thousand sentence pairs annotated for three-class entailment classification. The sentences come from image and video captions and were simplified and modified according to a set of transformation rules. Due to the small set of rules that were used to generate new sentences, SICK contains some phenomena that are unevenly distributed across entailment classes. We discuss additional properties of SICK in Chapter 4.

2.1.3 Stanford Natural Language Inference

The Stanford Natural Language Inference (SNLI) dataset [4] was created to train neural network models for textual entailment. It contains over 570,000 sentence pairs, each consisting of a premise sentence from an image caption dataset [9, 10], a hypothesis sentence written by a worker on Amazon Mechanical Turk, and a label for three-class entailment

classification. Table 5.6 contains a few examples.

The Mechanical Turk workers were provided with the premise sentence, and were asked to write one definitely true sentence, one possibly true sentence, and one definitely false sentence. The task design prompted workers to write hypothesis sentences that frequently parallel the premise in structure and vocabulary, and therefore the semantic relationships between premise and hypothesis are often limited to synonym/hyponym lexical substitution, replacement of short phrases, or exact word matching. As a result, several recent papers have demonstrated that, due to an uneven distribution of phenomena in the hypotheses, it is possible to correctly classify the majority of sentence pairs in SNLI without actually looking at the hypothesis [11, 12]. We discuss this further in Chapter 6.

2.2 APPROACHES

2.2.1 Feature-Based Models for Textual Entailment

Models for RTE were largely developed in the era of traditional NLP features, before distributional representations became widely used. Due to the difficulty of entailment problems in RTE, most approaches involve a complex pipeline of NLP tools. The first step is generally to extract a representation from the relevant sentences that will be used in the following inference step. A simple representation could involve lexical relationships between premise and hypothesis [13], identified via stemming and lemmatization preprocessing or the application of a lexical resource like WordNet. A more complex representation might be the structure produced by a syntactic parser, which could be used to identify common constituent phrases or dependency relations between the premise and hypothesis. Another potentially useful representation might be semantic role labels, which present a more abstract view of a sentence than a syntactic parse, and could be used to link sentences that don't share the same syntactic structure.

Given some representation of the premise and hypothesis, these models then use an inference step to judge whether the hypothesis follows from the premise or not. This decision could be based on any number of metrics. One possibility is to use alignment to link the premise and hypothesis: if enough of the hypothesis representation aligns to parts of the premise representation, then the model decides that the hypothesis is entailed [14]. A different metric to use would be *similarity*, which could be defined according to any number of similarity metrics, but in general would try to move beyond exact word matches between the premise and hypothesis [13, 15]. Yet another approach is to use a strict inference step, applying some kind of proof-based logical reasoning to the premise and hypothesis repre-

sentations [16, 17, 18, 19]. Although inference through proofs is a compelling approach, proof-based models are often brittle when applied to natural language applications.

2.2.2 The Addition of Distributional Representations

Many successful approaches to SICK use the same approach as RTE models, training a model with hand-engineered features. However, models for SICK often include distributional representations at the word or phrase level as additional features. The vast majority of submissions to the SICK SemEval shared task competition included a vector-based semantic representation. These systems included a wide range of other features as well, including topic models, word overlap, syntactic features, and other types of lexical similarity [20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31].

A few approaches to SICK have focused on the application of logical representations to textual entailment. Bjerva et al. [23] proposed a formal semantics-based model that uses logical inference to predict entailment. Beltagy et al. [32] combined a logic-based representation with a distributional approach to achieve a new state-of-the-art result on SICK. Although SICK is generally considered too small to train neural network models, a few recent works have applied convolutional neural networks [33] or GRUs [34] to textual entailment classification and established a new state of the art.

In Chapter 4, we will describe our submission to the SICK shared task competition, which involves applying denotational similarity to textual entailment [35].

2.2.3 Transition to Neural Networks

As SNLI generally does not contain named entities or other characteristics that might encourage the use of other NLP tasks as a preprocessing step, most approaches to SNLI have been end-to-end neural network models. These models do not involve an explicit feature extraction step where a model extracts a pre-defined representation (e.g. syntactic parsing, named entity recognition, or semantic role labeling) for each entailment item. Instead, most models start with a dense representation for each word in the sentence, train a neural network model that combines those word vectors into a sentence (or sentence pair) representation, and use the final representation as input to an entailment classifier (e.g. a multilayer perceptron).

Bowman et al. [4] initially illustrated the effectiveness of sequence-based neural networks with a model that applies an RNN [36] with an LSTM cell [37] to produce a sentence vector representation for each premise and hypothesis in SNLI. Although Bowman et al.'s initial neural network model achieved only comparable accuracy to a simple classifier trained on

unigram and bigram features, the neural entailment models that followed have far outstripped its performance. These models have mostly taken the same approach, producing dense sentence encodings as an intermediate step for entailment classification. Many use neural sequence models such as LSTMs [38, 39, 40, 41, 42], while others involve tree-based representations [43, 44, 45, 41].

Another line of work applies neural attention in order to reweight the most relevant parts of the premise and hypothesis sentences to improve entailment classification [46]. In some cases, this approach only relies on minimal word ordering information and does not require the full sentence to be passed through a sequence model [47].

In general, the architecture of these models could be easily applied to other sentence pair classification tasks, and is not inherently constructed to model phenomena specific to textual entailment alone. One downside of this is that there is often little discussion of what these models are actually learning, though there have been some recent efforts to go beyond reporting overall model accuracy [48]. On the other hand, these generic architectures may produce sentence embeddings that are useful for other semantic tasks [40].

This thesis in some respects mirrors the progression of textual entailment models described here. In Chapter 4 we describe a model for SICK that uses handcrafted features including distributional vector representations. Following that model, we moved to develop neural embedding models for entailment (Chapter 5 and Chapter 7). We will evaluate these models on some of the datasets described in this chapter, while also seeking to go beyond them and develop new, more challenging forms of evaluation that can tease out the differences between different semantic representations.

CHAPTER 3: THE DENOTATION GRAPH

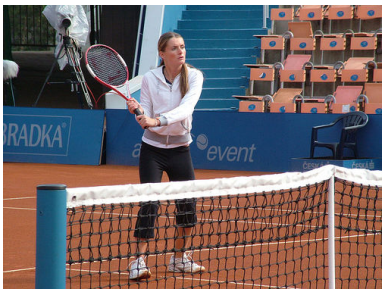
Part of understanding language is the ability to connect language to the external world. In addition to textual inference – identifying whether one piece of text implies the truth of another – this can involve linking language to visual representations. This idea is the motivation behind the denotation graph we introduced in Young et al. [9]. We constructed the denotation graph from FLICKR30K, an image-caption dataset that captures the range of linguistic expressions that can be used to describe a single scene. The denotation graph uses these image captions to generate a hierarchy of new descriptions that can be applied to existing images. We define a new similarity metric, *denotational similarity*, that expresses how often two captions in this hierarchy describe the same scene.

3.1 THE FLICKR30K IMAGE CAPTION CORPUS

FLICKR30K is a corpus of 31,783 images of everyday scenes collected from Flickr[1] and 158,915 captions we collected via crowdsourcing (Figure 3.1). The corpus extends the FLICKR8K dataset of Hodosh et al. [49]. Following their approach to collecting images and captions, we asked 5 different annotators on the crowdsourcing platform CrowdFlower to write a one-sentence description of each image. Since the annotators are not familiar with the images, they write captions that describe what is happening in the scene, e.g. *"Three people are setting up a tent,"* rather than the ungrounded personal captions that people write for their own photos (*"Our trip to the Olympic Peninsula"*). Annotators also use a range of specificity, so that a person playing a violin is described as both *performing a musical piece* and *bowing on a violin* by different annotators. The variety of descriptions allows us to identify phrases that are closely related in terms of their semantic grounding even though they are not easily linked by surface-level syntactic rewrite rules.

Work in this chapter was first published in P. Young, A. Lai, M. Hodosh, and J. Hockenmaier (2014), "From Image Descriptions to Visual Denotations: New similarity metrics for semantic inference over event descriptions," *Transactions of the Association of Computational Linguistics*, 2, 67–78 [9]. It is reprinted here with the permission of the copyright holder.

[1] `https://www.flickr.com/`



In an athletic stance, the woman awaits the tennis ball.
A girl wearing a white jacket about to swing a tennis racket.
A woman waits to hit a tennis ball over a net in front of an empty arena.
Woman tennis player holding a red and white tennis racket wearing capri pants.
A woman wearing a hooded sweatshirt and black pants with a ponytail playing tennis.

Four basketball players in action.
Young men playing basketball in a competition.
Four men playing basketball, two from each team.
Two boys in green and white uniforms play basketball with two boys in blue and white uniforms.
A player from the white and green high school team dribbles down court defended by a player from the other team.

Figure 3.1: Two images from FLICKR30K with their five descriptive captions.

3.2 DEFINING THE DENOTATION GRAPH

3.2.1 Denotational Semantics

Truth-conditional semantics is a theory of meaning that proposes that the meaning of a natural language sentence is the set of conditions under which the sentence is true. Under this theory, sentences s_1 and s_2 have the same meaning if their truth-conditions specify that s_1 and s_2 are always true under the exact same circumstances. For example, *"The man bought an expensive jacket"* and *"The man bought a pricy coat"* have the same truth conditions.

We can define entailment in these terms as well: if s_1 entails s_2, then for every situation in which s_1 is true, s_2 is true as well. For example, *"A woman swings her racket to hit the tennis ball over the net"* entails *"A woman is playing tennis."*

Another way to discuss this theory is to say that the meaning of a sentence, or its *denotation*, is the set of possible worlds or situations in which the sentence is true [50, 51, 52]. We can then compare the meaning of two sentences by comparing their denotations: if the set of possible worlds where s_1 is true is the same as the set of possible worlds where s_2 is true, then s_1 and s_2 have the same meaning; they are synonymous. Similarly, if s_2 is true in every possible world where s_1 is true, then s_1 implies or entails s_2.

3.2.2 Visual Denotation of Image Captions

We propose to instantiate denotational semantics' abstract notion of possible worlds with concrete sets of images, specifically the FLICKR30K images. On the language side, we consider the visually descriptive FLICKR30K captions that are linked to these images. The interpretation function $[\![\cdot]\!]$ maps a sentence s to its *visual denotation* $[\![s]\!]$, which we define to be the set of images $i \in U$, the universe of possible worlds, that s describes:

$$[\![s]\!] = \{i \in U | s \text{ is a truthful description of i}\} \tag{3.1}$$

Similarly, we map nouns and noun phrases to the set of images that depict those objects, and verbs and verb phrases to the set of images that depict those events. For example, the denotation of the phrase *man* is the set of images in the corpus that show a man, and the denotation of the phrase *person is rock climbing* is the set of images that depict a person rock climbing.

We can easily identify entailment relations via a partial ordering over descriptions: if s (e.g. *"a poodle runs on the beach"*) entails s' (e.g. *"a dog runs"*), then the denotation of s is a subset of the denotation of s' ($[\![s]\!] \subseteq [\![s']\!]$). We can say that the more general phrase s' subsumes the more specific s.

However, while a few FLICKR30K captions may naturally subsume other captions, but this is not true of most of the corpus. For example, annotators may have written captions like *"A poodle is running"* and *"A dog is running"* about the same image, but not the caption *"An animal is running."* Therefore, in order to produce a more complete subsumption hierarchy, we generate more generic descriptions from the caption sentences by applying several simple syntactic and lexical operations that preserve upward entailment. These operations preserve the truth of the original description s such that the resulting more generic description s' is entailed by s.

The result is the *denotation graph*, a subsumption hierarchy over phrases (see Figure 3.2). Each node in the denotation graph corresponds to a phrase s, which has an associated denotation $[\![s]\!]$, the set of images that correspond to the original captions from which this phrase can be derived. Any phrase in the graph entails its parent phrases, so the denotation of a node (e.g. *woman jog on beach*) is always a subset of the denotation of any of its ancestors (e.g. *woman jog*, *person jog*, *jog on beach*, or *beach*).

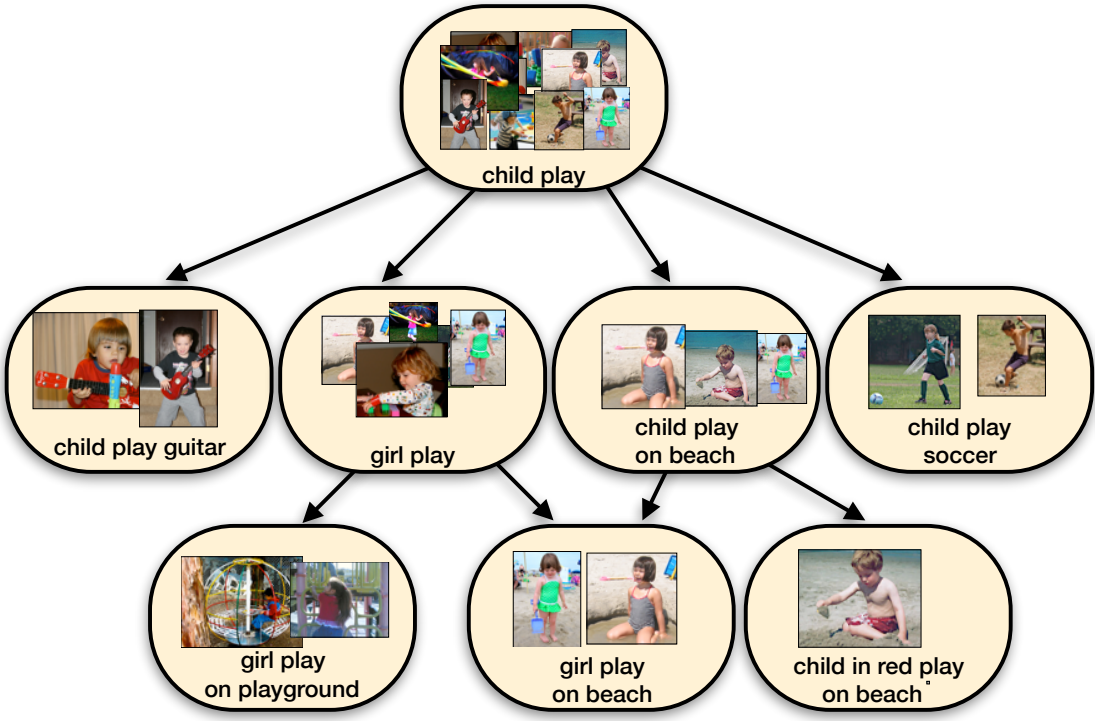


Figure 3.2: The denotation graph is a subsumption hierarchy over phrases associated with images.

3.2.3 Denotational Similarities

We can use visual denotations to measure the symmetric similarity of two phrases or the conditional probability of one phrase given another. We can quantify these relationships even between phrases that are not directly connected by edges in the graph.

The denotational probability of a phrase s in the denotation graph, $P_{[\![]\!]}(s; \text{graph})$, is a Bernoulli random variable that corresponds to the probability that a randomly drawn image, out of all the images used to generate the graph, can be described by s. For simplicity, we refer to this probability as $P_{[\![]\!]}(s)$ in this thesis. Given a denotation graph over N images, we can compute the denotational probability of an expression s with a denotation of size $|[\![s]\!]|$ as:

$$P_{[\![]\!]}(s) = \frac{|[\![s]\!]|}{N} \tag{3.2}$$

The joint probability of two expressions x and y expresses how likely it is that a situation can be described by both x and y:

$$P_{[\![]\!]}(x, y) = \frac{|[\![x]\!] \cap [\![y]\!]|}{N} \tag{3.3}$$

We can define the conditional probability $P_{[\![\,]\!]}(x|y)$ as the probability that a scene can be described by x if we know that y is a valid description of the same scene:

$$P_{[\![\,]\!]}(x|y) = \frac{P_{[\![\,]\!]}(x,y)}{P_{[\![\,]\!]}(y)} \tag{3.4}$$

We can also compute the normalized pointwise mutual information (PMI) between $[\![x]\!]$ and $[\![y]\!]$:

$$\text{pmi}_{[\![\,]\!]}(x,y) = \frac{\log\left(\frac{P_{[\![\,]\!]}(x,y)}{P_{[\![\,]\!]}(x)P_{[\![\,]\!]}(y)}\right)}{-\log\left(P_{[\![\,]\!]}(x,y)\right)} \tag{3.5}$$

Both $P_{[\![\,]\!]}(x|y)$ and $\text{pmi}_{[\![\,]\!]}(x,y)$ are similarity metrics that express the denotational overlap between two phrases: the likelihood that x and y are both reasonable description of the same scene. In this thesis, we will use the term *denotational similarity* somewhat indiscriminately to refer to either $P_{[\![\,]\!]}(x|y)$ or $\text{pmi}_{[\![\,]\!]}(x,y)$, but we will clearly state which metric we are using for each experiment.

3.3 CONSTRUCTING THE DENOTATION GRAPH

The construction of the denotation graph consists of the following steps: preprocessing and linguistic analysis of the captions, identification of applicable transformations, and repeated application of those rules to each caption in order to generate the graph itself. This section contains a high-level description of these steps; see Young's thesis [53] for additional details. The code to generate a denotation graph from a set of captions, including the modifications we describe in Section 3.3.6, is available at `https://github.com/aylai/DenotationGraph`.

3.3.1 Preprocessing

We apply the Linux spell checker; the OpenNLP tokenizer, POS tagger, and chunker (`http://opennlp.apache.org`); and the Malt parser [54] as preprocessing steps for graph generation. Since the vocabulary of FLICKR30K captions differs significantly from the data these tools are trained on, we resort to a number of heuristics to improve the resulting analyses.

After spell-checking, we normalize some common words and compounds that have multiple spelling variations, e.g. *barbecue (barbeque, BBQ)*, *gray (grey)*, *waterski (water ski)*, *brown-haired (brown haired)*.

Next, we tokenize the captions using the OpenNLP tokenizer. The OpenNLP POS tagger makes a number of systematic errors on our corpus, such as mis-tagging verbs as nouns. We correct these errors by applying a few deterministic rules (e.g. *climbs* is never a noun in our corpus, *stand* is a noun if it is preceded by *vegetable* but a verb when preceded by a noun that refers to people). These fixes apply to 17% of the data (27,784 captions).

We use the OpenNLP chunker to produce a shallow parse for each caption. Again, we apply some systematic fixes to address errors in 28,587 captions. We identify heads, determiners and prenominal modifiers in each NP chunk. The head may consist of more than one token if WordNet (or our hypernym lexicon, described below) contains an entry corresponding to the whole phrase (e.g. *little girl*). Finally, we use the Malt parser to identify subject-verb-object dependencies.

Since some heuristics require us to identify entity types, we also developed a lexicon of the most common entity types in our domain (people, clothing, bodily appearance (e.g. hair or body parts), containers of liquids, food items, and vehicles). The process of developing this lexicon is described in Young's thesis [53] (Section 2.4.8 and Section 3.5).

3.3.2 Hypernym Lexicon

We use WordNet [55] to construct a hypernym lexicon that allows us to replace head nouns with more generic terms. We only consider hypernyms that occur with sufficient frequency in the original captions (valid hypernyms of *adult* include *person* but not *organism*). We consider a single WordNet sense for each noun across the whole corpus, which works reasonably well due to the concreteness of the nouns in our corpus. In order to identify the correct WordNet sense, we use the heuristic cross-caption coreference algorithm of Hodosh et al. [56] to identify coreferent NP chunks among the original five captions of each image, and then apply a greedy voting algorithm to map each mention to a single WordNet sense. The sense determines the relevant hypernyms for that noun.

3.3.3 Caption Normalization

In order to increase the recall of the denotations we capture, we drop all punctuation marks and lemmatize nouns, verbs, and adjectives that end in *-ed* or *-ing* before generating the denotation graph.

We include a few normalization exceptions in order to distinguish between frequently occurring homonyms where the noun is unrelated to the verb. We change all forms of the verb *dress* to *dressed*, all forms of the verb *stand* to *standing* and all forms of the verb *park*

to *parking*.

Finally, we drop sentence-initial *there/here/this is/are* (as in *"there is a dog splashing in the water"*), and normalize the expressions *in X* and *dressed (up) in X* (where *X* is an article of clothing or a color) to *wear X*. We allow plural determiners *two* and *three*, and reduce anything higher to *some*. We also drop all singular determiners except *no*.

3.3.4 Transformation Rules

We define a set of transformation rules: syntactic and lexical operations that produce a more generic description s' given some string s. These rules can drop optional material, extract simpler constituents, or perform lexical substitutions. The resulting generic description should still correctly describe the image about which the original caption was written. These rules are used to create edges in the graph between s and s'. All transformation rules are repeatedly applied to each caption to obtain the simplest possible phrases, which are the root nodes in the denotation graph.

Drop Articles

We drop articles in all noun phrases:

$$\textit{the woman} \rightarrow \textit{woman}$$

We do not drop articles *no* (e.g. from *man wearing no shirt*) or *each* (e.g from *each other*).

Drop Noun Phrase Modifiers

We drop modifiers from noun phrases:

$$\textit{red shirt} \rightarrow \textit{shirt}$$

When there are multiple modifiers for a single head in the form $\texttt{adj}_1$ $\texttt{adj}_2$ `noun`, we drop each modifier separately only if the strings $\texttt{adj}_1$ `noun` and $\texttt{adj}_2$ `noun` both occur elsewhere in the corpus. This produces *white building* and *stone building* from the noun phrase *white stone building*. However, given the noun phrase *ice hockey player*, this rule will produce only *hockey player* and not *ice player*.

Replace Nouns with Hypernyms

We use our hypernym lexicon to replace noun phrase heads with more generic terms:

$$\textit{red shirt} \rightarrow \textit{red clothing}$$

When a single noun has more than one hypernym, we apply both transformations, which results in two parent nodes that each have an edge to a shared child node:

$$\textit{man} \rightarrow \textit{male}$$
$$\textit{man} \rightarrow \textit{adult}$$

We only allow the hypernym transformation after any age-based modifiers have been removed: *toddler* can be replaced with *child*, but not *older toddler* with *older child*.

Drop Other Modifiers

We drop adverbs from VP chunks as well as ADVP chunks:

$$\textit{run quickly} \rightarrow \textit{run}$$

We also drop prepositional phrases (a preposition followed by a possibly conjoined NP chunk) under several conditions. The preposition must be locational (*in*, *on*, *above*, etc.), directional (*towards*, *through*, *across*, etc.), or instrumental (*by*, *for*, *with*).

$$\textit{walk on the sidewalk} \rightarrow \textit{walk}$$

If the preposition is part of a phrasal verb (according to a predefined corpus-specific list), then we drop only the preposition or both the preposition and the direct object:

$$\textit{climb up a mountain} \rightarrow \textit{climb a mountain}$$
$$\textit{walk down a street} \rightarrow \textit{walk}$$

We also drop all *wear* `NP` constructions:

$$\textit{man wearing red} \rightarrow \textit{man}$$

Handle "X of Y" Cases

For noun phrases that occur in the form *X of Y*, most can be replaced with either *X* or *Y* and remain a true description, as in the sentence *"A man holding a glass of beer"*:

$$\textit{glass of beer} \rightarrow \textit{glass}$$
$$\textit{glass of beer} \rightarrow \textit{beer}$$

There are some exceptions, such as the phrase *body of water*, which can only be replaced with *water*, and expressions in the form *a kind/type/sort of X*, which we replace with *X*.

Handle "X or Y" Cases

When two noun phrases are joined with *or*, we allow both phrases to replace the entire string:

$$\textit{man or woman} \rightarrow \textit{man}$$
$$\textit{man or woman} \rightarrow \textit{woman}$$

Handle $\mathbf{VP_1}$-to-$\mathbf{VP_2}$ Cases

We replace VPs of the form *X to Y* with both *X* and *Y* if *X* is a movement or posture:

$$\textit{jump to catch} \rightarrow \textit{jump}$$
$$\textit{jump to catch} \rightarrow \textit{catch}$$

Otherwise we distinguish between cases we can only replace with *X* (because *Y* has not started yet):

$$\textit{wait to jump} \rightarrow \textit{wait}$$

and those we can only replace with *Y*:

$$\textit{seem to jump} \rightarrow \textit{jump}$$

Extract Simpler Constituents

Any noun phrase or verb phrase also corresponds to a node in the graph (after dropping the rest of the phrase):

$$\textit{man standing with a backpack} \rightarrow \textit{man}$$
$$\textit{man standing with a backpack} \rightarrow \textit{standing}$$
$$\textit{man standing with a backpack} \rightarrow \textit{backpack}$$

Finally, we use the Malt dependencies to identify subject-verb-object chunks that correspond to simpler sentences:

$$\textit{man look up while hiking} \rightarrow \textit{man look up}$$
$$\textit{man look up while hiking} \rightarrow \textit{man hiking}$$

The resulting sentences will be further simplified by extracting noun phrases (*man*, which can be further simplified via hypernyms to *male*, *adult*, and *person*) and verb phrases (*look up*, *hiking*).

3.3.5 Graph Generation

The naive approach to graph generation would be to generate all possible strings for each caption. However, this would produce far more strings than could be processed in a reasonable amount of time, and most of these strings would have uninformative denotations consisting of a single image. To make graph generation tractable, we use a top-down algorithm that generates the graph from the most generic (root) nodes, and stops once we reach a node that has a singleton denotation (Figure 3.3).

We start by identifying the rules that can apply to each original caption and use them to reduce each caption c as much as possible (SimplifyCaption). The resulting maximally generic strings are the root nodes of the graph, and we add them to Captions[c], the list of strings produced by each caption. When a root node string r has been produced by two different captions, we add r to the queue of nodes to be expanded.

While the queue is not empty, we remove a new node to be expanded. Each node string s in the queue is associated with a particular caption c which has its own set of transformations (Rules[c]). ExpandNode produces a set of child node strings that result from all applicable transformations of s with respect to c. For example, expanding node *run* with respect to the caption *"A woman runs along the shore"* will produce the child nodes *woman run* and

```
function GenerateGraph(ImageCorpus)
    Queue ← ∅
    Captions ← ∅
    Edges ← ∅
    Rules ← ∅
    for all c ∈ ImageCorpus do
        Rules[c], RootNodes ← SimplifyCaption(c)
        for all r ∈ RootNodes do
            Captions[r] ← Captions[r] ∪ {r}
            if |Captions[r]| = 2 then
                for all c′ ∈ Captions[r] do
                    push(Queue, ⟨c′, r⟩)
            else if |Captions[r]| > 2 then
                push(Queue, ⟨c, r⟩
    while ¬ empty(Queue) do
        (c, s) ← pop(Queue)
        children ← ExpandNode((s, Rules[c]))
        for all s′ ∈ children do
            Captions[s′] ← Captions[s′] ∪ {s′}
            Edges ← Edges ∪ {⟨s, s′⟩}
            if |Captions[s′]| = 2 then
                for all c′ ∈ Captions[s′] do
                    push(Queue, ⟨c′, s′⟩)
            else if |Captions[s′]| > 2 then
                push(Queue, ⟨c′, s′⟩
    return Captions, Edges
```

Figure 3.3: Denotation graph generation algorithm

run along the shore. However, expanding *run* with respect to a different caption *"A horse runs in a race"* will produce a different set of child nodes: *horse run* and *run in a race*.

For each child string s', we record an edge between s and s', and add s' to our list of generated strings. If s' has been generated by at least two captions, we add it to the queue to be further expanded. If not, expansion will pause at this node until a second caption generates s'.

3.3.6 Changes to the Graph Generation Process

In the course of this thesis, we made some modifications to the first version of the graph generation algorithm as described by Young et al. [9]. Below, we describe the changes that comprise our updated algorithm.

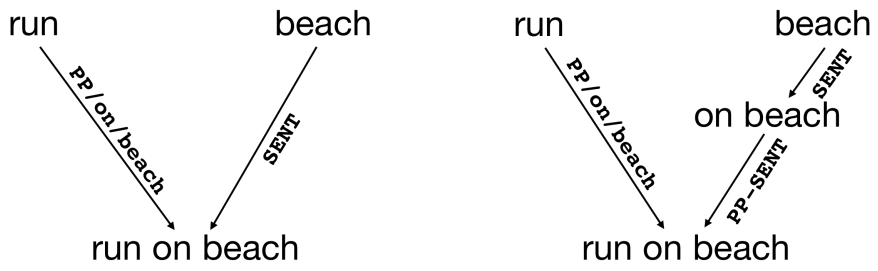


Figure 3.4: Left: the original denotation graph algorithm did not produce nodes corresponding to prepositional phrases. Right: we add prepositional phrase nodes as an intermediate node to the prepositional object node.

Adding Prepositional Phrase and Prepositional Object Nodes

The original graph algorithm allowed three types of nodes: entities (nouns or noun phrases), verb phrases (which may include direct objects), and sentences (all other nodes). For the purpose of the work in this thesis, we introduced prepositional phrase nodes, which can capture shared scene information, as shown in Figure 3.4.

Under the previous algorithm, a node *run on beach* was connected directly to the prepositional object *beach* via a `SENT` edge (which extracts nouns that are not subjects or direct objects), and there was no node for the phrase *on beach*. We added a fourth node type, prepositional phrases, to serve as an intermediate node connected to the prepositional object. So now, *run on beach* is a child node of PP node *on beach* via a new `PP-SENT` edge, and *on beach* is a child node of *beach* via a `SENT` edge. We reasoned that prepositional phrases often indicate salient scene information, and knowing that two images share the same prepositional phrase, not just the same noun phrase, provides additional signal.

Expanding the Allowed Constituents

The second change that we made to the graph generation process was to expand the types of constituents that can be extracted from a single sentence. The first version of the algorithm only allowed nodes corresponding to constituents as defined by the Penn Treebank annotation guidelines [57]. For a simple sentence consisting of `subj verb dobj`, we only generated parent nodes according to the rules `S` $\rightarrow$ `NP VP` and `VP` $\rightarrow$ `verb NP`. As a result, a node like *person eat pizza* has parent nodes *person* and *eat pizza*, and *eat pizza* has parent nodes *eat* and *pizza*. However, the subject and verb were not allowed to combine before the direct object was added to the verb, so the node *person eat* was disallowed for this sentence.

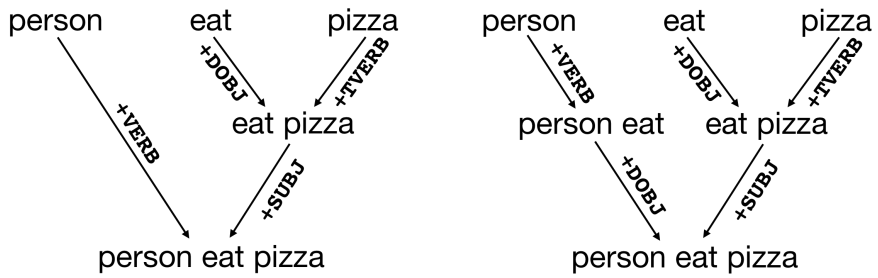


Figure 3.5: Left: the original denotation graph algorithm did not produce nodes corresponding `subj + verb` if the verb had a direct object. Right: we allow sentence nodes to drop the direct object, producing intermediate nodes like *person eat*.

We reasoned, however, that it is semantically informative in many cases to consider `subj + verb` as a constituent. For example, a new sentence *person eat pizza* may contain a direct object (*pizza*) that we have not seen before. However, we may still be able to say something about the denotational relationships between this sentence and other observed sentences about *people eating* if we can identify the common constituent *person eat*.

Therefore, we generate nodes that correspond to dropping the direct object to get `subj + verb` constituents. This results in a subgraph structure like the example in Figure 3.5.

Simplifying Transformation Rules

The original graph algorithm contained two types of rules: expansion rules and reduction rules. Under this approach, each caption started with a single maximally general sentence node, e.g. *person sit*. From this sentence node, reduction rules extracted the nouns and verbs from this sentence to produce parent nodes *person* and *sit*. In the other direction, expansion rules produced more specific strings in this caption, e.g. *adult sit*, *person in blue sit*, *person sit outside*.

Keeping track of the two directions of rewrite rules required a nontrivial amount of bookkeeping, so we modified the graph generation algorithm to express all transformation rules as expansions that produce a more specific phrase from a more general one. This simplifies the algorithm considerably. A side effect of this modification is that the expansion process for each caption now starts at a set of root nodes, all nouns or verbs, that constitute the most general phrases in the graph. So, rather than starting graph generation from a simple sentence, *person sit*, we start from two root nodes, *person* and *sit*, which both produce a shared child node *person sit* via expansion rules. All the strings for this caption can be produced starting from these two root nodes by expanding the graph downwards to more specific strings.

Denotation Size	# captions		Example strings
	Old	New	
$\|[\![s]\!]\| \geq 1000$	161	232	*person*, *ball*, *red shirt*, *group of adult*, *adult work*, *person play instrument*
$\|[\![s]\!]\| \geq 100$	1,921	2,887	*bicycle*, *guitar*, *mountain*, *dish*
$\|[\![s]\!]\| \geq 10$	22,683	33,940	*produce*, *pillar*, *adult with helmet*, *woman with blue shirt*
$\|[\![s]\!]\| \geq 5$	53,341	57,631	*white dog with brown spot*, *yacht*, *two scuba diver*, *mom with child*
$\|[\![s]\!]\| \geq 2$	230,811	259,712	*woman with short hair and clothing*, *two adult at work*, *pyrotechnics*, *squid*
$\|[\![s]\!]\| \geq 1$	1,749,096	2,889,013	*adult dressed for colder weather*, *bare pine table*, *rabbit costume*, *thick cord*

Table 3.1: Distribution of the denotation size of phrases in the FLICKR30K denotation graph. *Old* is the count from the original graph [9]. *New* is the count from the modified graph algorithm as described in this thesis.

3.4 PROPERTIES OF THE GRAPH AND RESULTING SIMILARITIES

3.4.1 Size and Coverage

The denotation graph contains 2,889,013 captions, 259,712 of which describe more than a single image. Table 3.1 shows how denotation size is distributed over the captions. *Old* refers to the graph generated by the original algorithm [9], while *New* refers to the graph generated by the algorithm described in this thesis. The new graph contains 232 captions that describe each over 1,000 images. In addition to general nouns such as *person*, these captions also include simple sentences such as *woman standing*, *adult work*, *person walk street*, or *person play instrument*.

Since the graph is derived from the original captions by simple syntactic operations, the denotations it captures are likely to be incomplete. For example, $[\![$*soccer player*$]\!]$ contains 251 images, $[\![$*play soccer*$]\!]$ contains 234 images, and $[\![$*soccer game*$]\!]$ contain 119 images. It is probably the case that most of the *soccer player* images involve a *soccer game* or *playing soccer*, but some of those associations are missing from the graph. We also have not attempted to identify word order variation (*stick tongue out* vs. *stick out tongue*) or context-specific equivalent prepositions (*look into mirror* vs. *look in mirror*), let alone synonyms. However, despite this brittleness, the graph expresses a large number of semantic associations.

3.4.2 Denotational Similarity Examples

Having generated a denotation graph from FLICKR30K captions, we can now compute two types of denotational similarities between any two phrases x and y: the pointwise mutual information $\text{pmi}_{[\![\,]\!]}(x, y)$ (Equation 3.5) and the conditional probability $P_{[\![\,]\!]}(x|y)$ (Equation 3.4). The following examples show that denotational similarities find events that are closely related by entailment, rather than merely topically related:

$P_{[\![\,]\!]}(x\|y)$	x	y
0.962	sit	eat lunch
0.846	play guitar	strum
0.811	surf	catch wave
0.800	ride horse	rope calf
0.700	listen	sit in classroom

If someone is *eating lunch*, it is likely that they are *sitting*, and people who *sit in a classroom* are likely to be *listening* to somebody. These entailments can be very precise: *walk up stair* entails *ascend*, but not *descend*; the reverse is true for *walk down stair*:

$P_{[\![\,]\!]}(x\|y)$	x = ascend	x = descend
y = walk up stair	0.32	0.0
y = walk down stair	0.0	0.31

Pointwise mutual information captures paraphrases as well as closely related events: people *look in a mirror* when *shaving their face*, and baseball players may *try to tag* someone who is *sliding into base*:

$\text{pmi}_{[\![\,]\!]}(x, y)$	x	y
0.835	open present	unwrap
0.826	lasso	try to rope
0.791	get ready to kick	run towards ball
0.785	try to tag	slide into base
0.777	shave face	look in mirror

We can compare denotational similarities to distributional similarities that are computed over the same corpus. (In Section 3.5.1, we explain how we compute distributional similarities on FLICKR30K data.) We look at the expressions that are most similar to *play baseball* or *play football* according to the denotational $\text{pmi}_{[\![\,]\!]}$ and the distributional similarities (Σ). Denotational similarity finds actions that are part of the same sport, while distributional similarity finds other high-level events that are similar to *play baseball* or *play football*:

play baseball

$\text{pmi}_{[\![]}$		Σ	
0.674	tag him	0.859	play softball
0.637	hold bat	0.782	play game
0.616	try to tag	0.768	play ball
0.569	slide into base	0.741	play catch
0.516	pitch ball	0.739	play cricket

play football

$\text{pmi}_{[\![]}$		Σ	
0.623	tackle person	0.826	play game
0.597	hold football	0.817	play rugby
0.545	run down field	0.811	play soccer
0.519	wear white jersey	0.796	play on field
0.487	avoid	0.773	play ball

3.5 EVALUATING DENOTATIONAL SIMILARITIES

In this section, we present two different evaluations comparing the utility of denotational and distributional similarities.

3.5.1 Approximate Textual Entailment

In order to evaluate the utility of denotational similarities, we apply them to an *approximate entailment task* (ATE). ATE is loosely modeled after the Recognizing Textual Entailment problem [2] and consists of deciding whether a simplified caption h (the hypothesis) can describe the same image as a set of four captions $P = \{p_1, ..., p_4\}$ (the premises).

Data

We generate ATE items $\langle P, h \rangle$ as follows: given an image, a random subset of four of its captions forms a set of premises. The hypothesis is either a short verb phrase or sentence corresponding to a node in the denotation graph, and may or may not come from the same image. By focusing on short hypotheses, we minimize the possibility that the hypotheses contain extraneous details that cannot be inferred from the premises. Positive examples are

generated by choosing a node h as the hypothesis and an image $i \in [\![h]\!]$ such that exactly one caption of i generates h. We use the other four captions of i as the set of premises (since they do not trivially entail h through the graph generation transformations, denotational approaches should not have an unfair advantage over distributional methods). Negative examples are generated by choosing a node h as the hypothesis and randomly selecting four of the captions of an image $i \notin [\![h]\!]$.

The training items are generated from the captions in the training split of FLICKR30K, and the test items are generated from the disjoint test split. The VP dataset contains 290,000 training items and 16,000 test items, while the S dataset contains 400,000 training items and 22,000 test items. In each set, half of the items are positive and half are negative.

Denotational Similarity Features

We compute denotational similarities over node pairs in a denotation graph generated from the FLICKR30K training split images. We only consider pairs of nodes $\langle n, n' \rangle$ if their denotations contain at least 10 images each and their intersection contains at least two images.

To map an item $\langle P, h \rangle$ to denotational similarity features, we represent the premises as the set of all nodes P that are ancestors of the premise captions. A sentential hypothesis is represented as a set of nodes $H = \{h_{\texttt{S}}, h_{\texttt{subj}}, h_{\texttt{VP}}, h_{\texttt{verb}}, h_{\texttt{dobj}}\}$ that correspond to the sentence as well as its subject, verb phrase, verb, and direct object. A VP hypothesis is represented by nodes $H = \{h_{\texttt{VP}}, h_{\texttt{verb}}, h_{\texttt{dobj}}\}$.

Given the premise and hypothesis node sets, we compute global similarity features over all constituent types as well as constituent-specific features. We compute both types of denotational similarity ($\text{pmi}_{[\![]\!]}(h, p)$ and $P_{[\![]\!]}(h|p)$) as our features. For the global features, we compute max and sum features over all node comparisons $\langle h, p \rangle$ such that for $p \in P$, $h \in H$:

$$\text{SUM} = \sum_{p,h} \text{sim}(h, p) \tag{3.6}$$

$$\text{MAX} = \max_{p,h} \text{sim}(h, p) \tag{3.7}$$

The constituent-specific features are computed similarly, taking the sum and max over all node comparisons $\langle h_x, p \rangle$ where h_x is a node in H of constituent type x. Finally, we include node-specific constituent-type features $sum_{x,s}$ and $max_{x,s}$ for each constituent string s.

Compositional Distributional Similarity Features

For distributional similarity features, we first compute word vectors on the FLICKR30K captions as well as over the British National Corpus (BNC) and Gigaword [58] corpus. For each corpus, we define the context words to be the 1000 most frequent words. Then we map each word w that appears at least 10 times in the corpus to a vector of the positive normalized pointwise mutual information scores between w and the context words.

We consider multiple definitions of context in order to provide a fairer comparison between denotational and distributional similarities. First, CAP vectors use a more traditional definition of distributional context: w occurs in the context of w' if they appear in the same sentence. IMG vectors are more comparable to denotational similarities: w and w' co-occur if they occur in any captions that describe the same image. HYP vectors are computed like IMG vectors after augmenting the FLICKR30K captions with hypernyms. Finally, ALL vectors include distributional vector similarity features computed on BNC and Gigaword.

We use two standard compositional baselines to combine the word vectors into a single sentence vector: addition ($s_\Sigma = w_1 + ... + w_n$, which can be interpreted as a disjunctive operation), and element-wise (Hadamard) multiplication ($s_\prod = w_1 \odot ... \odot w_n$, which can be viewed as a conjunctive operation). In both cases, we represent the premise set as the sum of all four caption vectors $p = p_1 + ... + p_4$. This gives two compositional similarity features: $\Sigma = \cos(p_\Sigma, h_\Sigma)$, and $\Pi = \cos(p_\Pi, h_\Pi)$.

Experimental Results

For each model, we train a binary logistic regression classifier with MALLET [59]. In addition to the features described above, each model contains bag-of-words features expressing the word overlap between the premises and the hypothesis (after expanding the premises with our hypernym lexicon).

Table 3.2 shows the test accuracy of our models on the VP and S node tasks. The denotational models solidly outperform all of the models trained on distributional features. For the distributional similarities, we observe that the accuracy of the distributional models tends to increase when we use the denotational definition of context (IMG), add hypernyms (HYP), and finally add information from other corpora (ALL). The HYP column shows that the denotational metrics clearly outperform any distributional metric when both have access to the same information. Although the distributional models benefit from the BNC and Gigaword-based similarities (ALL), their performance is still below that of the denotational models.

	VP task				S task			
	Cap	Img	Hyp	All	Cap	Img	Hyp	All
Distributional Π	68.4	70.5	70.5	70.3	75.3	76.6	77.1	77.3
Distributional Σ	67.8	71.4	71.6	71.4	76.9	78.1	79.1	79.2
Π, Σ	69.8	72.7	**72.9**	72.7	77.0	78.6	79.3	**79.6**
$\text{pmi}_{[\![]\!]}$			74.9				80.2	
$P_{[\![]\!]}$			73.8				79.5	
$\text{pmi}_{[\![]\!]}$, $P_{[\![]\!]}$			**75.5**				**81.2**	

Table 3.2: Test accuracy on Approximate Entailment. Denotational features ($\text{pmi}_{[\![]\!]}$, $P_{[\![]\!]}$) outperform distributional features (Π, Σ).

3.5.2 Semantic Textual Similarity

We also evaluate the effectiveness of denotational similarities on the SemEval 2012 Semantic Textual Similarity (STS) task [60], which contains 1500 sentence pairs from the MSR Video Description Corpus [61]. The goal of this task is to score the relatedness of two sentences from 0 (unrelated) to 5 (equivalent).

Denotational Similarity Features

Since the STS task is symmetric, we only consider $\text{pmi}_{[\![]\!]}$ similarities. Similar to the previous experiment, we again represent each sentence with five types of constituents. However, these sentences are longer than the hypotheses in ATE, so we can rarely map them to a single complete sentence node and they often contain other noun phrases in addition to the subject and object. Therefore, we replace the sentence node feature with a noun phrase constituent feature: $S = \{s_{\texttt{subj}}, s_{\texttt{VP}}, s_{\texttt{verb}}, s_{\texttt{dobj}}, s_{\texttt{NP}}\}$. Each constituent type may contain multiple nodes (especially the NP nodes), and is divided into two groups: nodes that appear in the original sentence, and ancestors of those nodes (which contain hypernym information or dropped modifiers, etc). We include features that compare constituents of the same type as well as constituents of any type. These denotational features are the same ones we use in Chapter 4; Section 4.2.3 contains more details as to how these features are computed.

Experimental Results

We use a state-of-the-art model, DKPro Similarity [62], which consists of a log-linear regression model trained on multiple text features (word and character n-grams, longest common substring and longest common subsequence, Explicit Semantic Analysis [63], and

Features	Pearson r
DKPro	0.868
$+\Sigma, \Pi$	0.880
$+\text{pmi}_{[\![]\!]}$	0.888
$+\Sigma, \Pi, \text{pmi}_{[\![]\!]}$	**0.890**

Table 3.3: Performance on the STS MSRvid task: DKPro Similarity plus compositional (Σ, Π) and/or denotational similarities ($\text{pmi}_{[\![]\!]}$) from FLICKR30K.

Resnik's WordNet-based similarity [64]). We investigate the effects of adding distributional and denotational similarity features to this system.

Table 3.3 shows experimental results for four models: We compare the off-the-shelf DKPro Similarity model to versions where we appended different FLICKR30K features. We either include the distributional features (Σ, Π) from Section 3.5.1, the constituent-based $\text{pmi}_{[\![]\!]}$ denotational features, or both. Models are evaluated according to the Pearson correlation r of their predicted similarity scores to the human-annotated ones. We see that the denotational similarities outperform the distributional similarities. Even after adding distributional similarity features, the bulk of the improvement comes from the denotational similarity features.

3.6 RELATED WORK

The most similar line of work to the denotation graph is Berant et al. [65]'s *entailment graph*. Entailment graphs contain nodes that correspond to propositional templates, a binary template where at least one of the two arguments is a variable (e.g. *X treats Y* or *X treats nausea*). An edge (u, v) in the entailment graph means that template u entails template v. For example, *X is diagnosed with asthma* entails *X suffers from asthma*. Historically, textual entailment systems required knowledge of entailment patterns, which specify an entailment relation between two fragments of text, and some of which can be encoded as these propositional templates. Entailment graphs have the potential to facilitate the acquisition of entailment rules for predicates while enforcing other relations between rules, e.g. transitivity of entailment.

Berant et al.'s goal is to learn focused entailment graphs, concerning a single target concept (e.g. *nausea*), from data. They present an approach that learns the edges of an entailment graph given a set of propositional templates. The propositional templates are extracted from a large corpus and labeled as positive or negative examples of entailment using WordNet hypernym information. These examples are used to train an entailment classifier, which

produces a score for each possible entailment graph edge (u, v). Finally, they apply global restraints to produce the optimal graph given the nodes and the potential edges.

The entailment graph resembles our denotation graph in that the nodes are connected by directed edges that indicate an entailment relationship. However, the denotation graph differs from the entailment graph in two ways. First, in addition to directed entailment edges, the denotation graph also expresses graded entailment (denotational conditional probability), which is defined extensionally in terms of the images at each node. This allows us to express that one phrase x is reasonably likely but not guaranteed to be true given y, based on the image overlap. Second, the nodes in the entailment graph correspond to generic propositional templates (*X treats Y*), while nodes in our denotation graph correspond to complete propositions (*a dog runs*).

Kotlerman et al. [66] extend entailment graphs to handle complete natural language texts instead of predicates, so they consider full propositions like our denotation graph does. These *textual entailment graphs* still differ from the denotation graph in that they lack the extensional image representation that allows us to express similarity between unconnected nodes in the graph. Kotlerman et al. present an evaluation for merging paraphrase or near-paraphrase nodes in their graph, which is a step that might prove useful for improving the denotation graph. However, they only present this evaluation on a very small dataset (29 graphs with a total of 756 nodes).

CHAPTER 4: A TEXTUAL ENTAILMENT MODEL WITH HANDCRAFTED DENOTATION FEATURES

In the previous chapter, we defined denotational similarities and applied them as features for an approximate textual entailment task. We now apply these denotational similarity features to a real textual entailment dataset. In this chapter, we describe the model we developed for the SemEval 2014 shared task [8] on textual entailment and semantic relatedness. Our model included denotational similarity features as well as other features based on distributional similarity and alignment. We use this model to compare the effectiveness of denotational features to other handcrafted features. We also analyze the SICK dataset and their approach to building a textual entailment dataset.

4.1 A DATASET FOR SEMANTIC SIMILARITY AND TEXTUAL ENTAILMENT

The Sentences Involving Compositional Knowledge (SICK) dataset [3] was constructed to evaluate textual entailment and semantic relatedness models without requiring the kind of world knowledge previously required for these tasks. SICK was used for the SemEval 2014 Task 1 competition to evaluate systems on textual entailment and semantic relatedness.

4.1.1 Dataset Construction

The SICK dataset contains 9927 English sentence pairs (4500 train, 500 development, 4927 test), each annotated with a semantic relatedness score and a textual entailment label. The authors started with sentences from an image description dataset, Flickr8K [49], and a video description dataset, the MSR Video Description Corpus subset of the SemEval 2012 STS dataset. They simplified and transformed these sentences according to a small set of syntactic and lexical transformation rules, and then randomly paired the resulting sentences to create sentence pairs. Each sentence pair is labeled with a textual entailment relation and a semantic relatedness score.

Work in this chapter was first published in A. Lai and J. Hockenmaier (2014), "Illinois-LH: A Denotational and Distributional Approach to Semantics," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 329–334 [35]. It is reprinted here with the permission of the copyright holder.

Premise	Hypothesis	TE	SR
A man is jumping into an empty pool.	There is no biker jumping in the air.	N	1.2
Two angels are making snow on the lying children.	Two children are lying in the snow and are making snow angels.	N	2.9
The young boys are playing outdoors and the man is smiling nearby.	There is no boy playing outdoors and there is no man smiling .	C	3.6
The brown horse is near a red barrel at the rodeo.	The brown horse is far from a red barrel at the rodeo.	C	3.6
A person is riding the bicycle on one wheel.	A man in a black jacket is doing tricks on a motorbike.	N	3.7
Two groups of people are playing football.	Two teams are competing in a football match.	E	4.7
A person in a black jacket is doing tricks on a motorbike.	A man in a black jacket is doing tricks on a motorbike.	E	4.9

Table 4.1: SICK sentence pair examples with gold textual entailment and semantic relatedness labels.

4.1.2 Annotation

Each sentence pair in SICK was annotated by multiple non-expert annotators via a CrowdFlower crowdsourcing task. Table 4.1 contains some examples.

Textual entailment The entailment class label is the majority of five human judgments. Annotators labeled each sentence pair as ENTAILMENT, CONTRADICTION, or NEUTRAL to indicate whether sentence B is true, false, or neither given the information in sentence A. Models are evaluated according to overall accuracy.

Semantic relatedness The goal of semantic relatedness, also known as semantic textual similarity [60], is to score the relatedness of two texts on a continuous scale. In contrast to a classification task like paraphrase recognition or textual entailment, semantic relatedness can express on a graded scale that sentences are more or less topically related. In SICK, the semantic relatedness score is a real number between 1 and 5 that is the mean of 10 human ratings. A score of 1 means that the sentences are completely unrelated and a score of 5 means that the sentences are very related, perhaps even paraphrases. Models are evaluated using the Pearson correlation between the predicted and gold scores.

4.2 SENTENCE PAIR COMPARISON FEATURES

Our model combines various sources of semantic similarity, including distributional similarity features, denotational similarity features, and alignment features based on shallow syntactic structure, to model both semantic relatedness and textual entailment. In this section, we define these features.

4.2.1 Negation

In SICK, the contradiction class is often marked by explicit negation, for example:

PREMISE: The man is stirring the sauce for the chicken.
HYPOTHESIS: The man is not stirring the sauce for the chicken.

⇒ CONTRADICTION

We define a binary feature that indicates whether either sentence contains *not*, *no*, or *nobody*.

4.2.2 Word Overlap

We define a word overlap feature:

$$\text{overlap} = \frac{|W_1 \cap W_2|}{\frac{1}{2}\left(len\left(s_1\right) + len\left(s_2\right)\right)} \tag{4.1}$$

where W_i is the set of word types that appear in sentence s_i, and the denominator is the average sentence length. We compute the word overlap after lemmatizing the sentences and removing stopwords.

4.2.3 Denotational Constituent Similarity

As we stated in Chapter 3, denotational similarity is intended to capture entailment-like relations between events. We include denotational similarity features in our model to evaluate that hypothesis. As with the STS-2012 task (Section 3.5.2), we cannot expect that any given sentence in SICK will have been observed in FLICKR30K or the resulting denotation graph. Therefore, we decompose each sentence into constituent phrases and use denotational similarity to compare multiple phrases between the premise and the hypothesis.

We identify five types of constituents for each sentence: subject noun phrases, verbs, verb phrases, direct object noun phrases, and other noun phrases. Following the same procedure

Constituent Decomposition

s_1 = *A player is throwing the ball.*

subj(s_1) = {*player*}	anc(subj(s_1)) = {*person*}
vp(s_1) = {*throw ball*}	anc(vp(s_1)) = {*throw*}
verb(s_1) = {*throw*}	
dobj(s_1) = {*ball*}	

s_2 = *Two teams are competing in a football match.*

subj(s_2) = {*two team*}	anc(subj(s_2)) = {*team*}
vp(s_2) = {*compete in football match*}	anc(vp(s_2)) = {*compete*, *compete in match*}
verb(s_2) = {*compete*}	
dobj(s_2) = {*football match*}	anc(dobj(s_2)) = {*match*}

Feature	Constituent Type	s_1 constituent	s_2 constituent
leaf_sim	subj	*player*	*two team*
	verb	*throw*	*compete*
	dobj	*ball*	*football match*
mix_sim	subj	*person*	*two team*
	vp	*throw*	*compete in football match*
anc_sim	subj	*person*	*team*
	verb	*throw*	*compete*
all_sim	–	*player*	*compete in match*
	–	*throw ball*	*football match*
	–	*ball*	*team*

Table 4.2: Examples of constituent comparisons based on the feature template applied to the specified sentence pair. This sentence pair decomposition method is used for both the denotational and distributional features.

we used to generate the graph, we use the Malt parser to identify subject-verb-object dependencies and extract constituents. We then attempt to map each constituent phrase to a node in the denotation graph. When the constituent phrase exists in the graph (because it was generated from the FLICKR30K captions), we can compute the denotational similarity between this constituent phrase and any other node in the graph. For each constituent type t, we also define anc(t), which is the set of parents and grandparents of all phrases t in the denotation graph. The top half of Table 4.2 shows two sentences from SICK with their corresponding constituents and example constituent comparisons.

However, phrases that are unique to SICK have no corresponding node in the pre-computed graph and therefore no quantifiable similarity to other phrases. This affects the coverage of the denotational features, which we discuss in Section 4.4.2.

We define two sets of denotational similarity features: *typed* and *untyped*. Both sets express how often we expect the specified constituents to apply to the same situation. The

typed features compare constituents of the same type, while the untyped features compare constituents of any type. The bottom half of Table 4.2 shows examples of constituent comparisons.

We specify $t(s)$ to be the set of phrases of constituent type t in sentence s. $t_i(s)$ specifies a single phrase in this set (ancestor phrase sets in particular often contain more than one phrase). We define three kinds of typed similarity features which are parameterized by the constituent type $t \in \{\text{SUBJ, VP, VERB, DOBJ, NP}\}$:

$$\text{LEAFSIM}_t = \max_{i,j}(\text{pmi}_{[\![]\!]}(t_i(s_1), t_j(s_2))) \tag{4.2}$$

$$\text{MIXSIM}_t = \max(\max_{i,j}(\text{pmi}_{[\![]\!]}(t_i(s_1), anc(t_j(s_2)))), \max_{i,j}(\text{pmi}_{[\![]\!]}(anc(t_i(s_1)), t_j(s_2))) \;\;) \tag{4.3}$$

$$\text{ANCSIM}_t = \max_{i,j}(\text{pmi}_{[\![]\!]}(anc(t_i(s_1)), anc(t_j(s_2))) \tag{4.4}$$

The LEAFSIM features measure the denotational similarity between constituents of the same type. The MIXSIM features measure the denotational similarity between constituents in one sentence with the constituent ancestors of the same type in the other sentence. ANCSIM features measure the denotational similarity between constituent ancestors of the same type. For all denotational similarity features, we use the normalized pointwise mutual information $\text{pmi}_{[\![]\!]}(x, y)$ between phrases x and y in the denotation graph (Equation 3.5). We found in previous experiments that $\text{pmi}_{[\![]\!]}(x, y)$ tended to be more robust to new data.

We additionally define three kinds of untyped constituent similarity features, again parameterized by the constituent type t, where $all(s)$ specifies constituent phrases of any type in sentence s:

$$\text{ALLMAX}_t = \max \quad (\max_{i,j}(\text{pmi}_{[\![]\!]}(t_i(s_1), all_j(s_2))) \quad , \quad \max_{i,j}(\text{pmi}_{[\![]\!]}(all_i(s_1), t_j(s_2)))) \tag{4.5}$$

$$\text{ALLMIN}_t = \min \quad (\max_{i,j}(\text{pmi}_{[\![]\!]}(t_i(s_1), all_j(s_2))) \quad , \quad \max_{i,j}(\text{pmi}_{[\![]\!]}(all_i(s_1), t_j(s_2)))) \tag{4.6}$$

$$\text{ALLSUM}_t = \text{sum} \quad (\max_{i,j}(\text{pmi}_{[\![]\!]}(t_i(s_1), all_j(s_2))) \quad , \quad \max_{i,j}(\text{pmi}_{[\![]\!]}(all_i(s_1), t_j(s_2)))) \tag{4.7}$$

4.2.4 Distributional Constituent Similarity

While denotational similarity captures the degree to which one phrase entails another, its coverage is limited to phrases that occur frequently in the denotation graph. To alleviate the sparsity of the denotational similarity features, we include distributional vector-based similarity features over the same constituents.

As one point of comparison regarding coverage, for constituent features comparing subject pairs, we have non-zero distributional similarity for 87% of instances in the development data, but non-zero denotational similarity for only 56% of the same instances. For example, the phrases *football* and *compete* have a denotational overlap of only four images in the training data, which is below the threshold (10 images) that we use to define denotational similarity. As a result, *football* and *compete* have a denotational similarity of 0. On the other hand, we have distributional vectors for *football* and for *compete* which we can use to compute a non-zero distributional similarity.

For the distributional similarity feature, we start by computing count-based distributional co-occurrence vectors over the FLICKR30K captions (after lemmatizing and removing stopwords). We compute vectors for the 3254 tokens that appear at least 10 times in FLICKR30K. We define the vector space using the 1000 most frequent lemmas in FLICKR30K as context words. For each word w_i in our vocabulary, we compute a vector where the jth entry is the pointwise normalized PMI between target w_i and context word w_j:

$$\text{pnPMI}(w_i, w_j) = \max\left(0, \frac{\log\left(\frac{P(w_i,w_j)}{P(w_i)P(w_j)}\right)}{-\log\left(P\left(w_i, w_j\right)\right)}\right) \tag{4.8}$$

$P(w_i)$ is the fraction of images with at least one caption containing target word w_i, and $P(w_i, w_j)$ is the fraction of images whose captions contain both w_i and w_j. Following work that extends distributional similarity to phrases [67], we use element-wise multiplication to compose word vectors $w_1 ... w_n$ into a phrase vector p:

$$p = w_1 \odot ... \odot w_n \tag{4.9}$$

where $\odot$ is the Hadamard (element-wise) multiplication of corresponding vector components, i.e. $p_i = u_i \cdot v_i$.

The resulting vectors have a somewhat broader definition of context than typical distributional count vectors. Rather than defining the context of a target word to be within a fixed-size window of k words to either side or even within the same sentence as the target word, our definition of context is whether the context word and the target word appear

together in the same set of five captions that describe the same image. This modified distributional representation is slightly closer to denotational similarity, as it can take advantage of the shared denotational information across captions that share an image.

We define typed and untyped distributional similarity features that mirror the denotational constituent-based features described in Section 4.2.3. We simply replace the denotational definition of $\text{pmi}_{[]}(a, b)$ with the cosine similarity between constituent phrase vectors a and b, where a and b are the pointwise multiplication product of their word vectors.

4.2.5 Alignment

In many semantic tasks involving sentence pairs, it can be useful to identify aligned phrases between the sentences and compute features over the alignments. For SICK in particular, both CONTRADICTION and ENTAILMENT pairs often have a premise and hypothesis with similar or identical syntactic structure. Therefore, it can be useful to identify the parts of the sentences that align, and compute features over the remaining words.

We start by identifying the longest subsequence of matching lemmas between the premise and the hypothesis. We then compute *word alignment* features from the aligned subsequence and *phrase similarity* features from the remaining words in both sentences. We use the Needleman-Wunsch algorithm [68] to compute longest subsequence of matching words between the lemmatized premise and the lemmatized hypothesis. To do this, we define the similarity between two lemmas to be 1.0 if the words are identical and 0.0 otherwise, and we do not penalize gaps.

The algorithm produces an alignment between the two sentences and a set of unaligned tokens for each sentence. In the following example, the underlined parts of both sentences constitute the longest subsequence alignment.

PREMISE: A brown and white dog is running through the tall grass.
HYPOTHESIS: A brown and white dog is moving through the wild grass.

⇒ ENTAILMENT

This results in a set of (lemmatized) tokens from each sentence that are not contained in the longest alignment:

PREMISE: run, tall
HYPOTHESIS: move, wild

We will use alignment and unaligned tokens to compute the alignment features below.

Token Alignment

Based on the aligned subsequences and remaining unaligned tokens, we compute the following features:

- Number of words in the premise and in the hypothesis
- Ratio of the number of words in the premise to the number of words in the hypothesis
- Number of words in the longest aligned subsequence
- Maximum, minimum, and average unaligned phrase lengths in the premise
- Maximum, minimum, and average unaligned phrase lengths in the hypotheses
- Maximum, minimum, and average unaligned phrase lengths in the premise and the hypothesis

Phrase Similarity

We compute features to express the similarity of the remaining unaligned phrases: when two sentences have a large overlap, their differences can be very informative. If only a single word in the premise differs from the hypothesis, the similarity of those two words can determine the relationship between the sentences. If the words are synonyms, then the premise entails the hypothesis. If the words are antonyms, then the hypothesis contradicts the premise.

We use the same distributional similarity metric defined in Section 4.2.4: we compose phrase vectors as the element-wise product of FLICKR30K word vectors. We consider all unaligned phrases between the premise and the hypothesis and greedily pair them according to the highest cosine similarity between phrase vectors until no candidate pairs remain. We run this algorithm twice: once for *simple* similarity where any two phrases are a valid candidate pair, and once for *strict* similarity where two phrases are a valid candidate pair only if they have the same shallow parse label.

From our previous "*brown and white dog*" example, strict similarity produces only one set of valid pairs:

$$\langle \text{ run}_{\texttt{VP}}, \text{ move}_{\texttt{VP}} \rangle$$
$$\langle \text{ tall}_{\texttt{NP}}, \text{ wild}_{\texttt{NP}} \rangle$$

Simple matching could produce two possible sets (⟨run, tall⟩ and ⟨move, wild⟩, or ⟨run, wild⟩ and ⟨move, tall⟩), depending on which pair has the highest distributional similarity.

Given the pairs of similar phrases, we compute the following features for both simple and strict similarity:

- Number of unaligned phrases in the premise and in the hypothesis
- Ratio of the number of unaligned phrases in the premise to the number of unaligned phrases in the hypothesis
- Maximum, minimum, and average similarity of paired phrases
- Maximum, minimum, and average word lengths of paired phrases
- Number of matched phrases
- Number of crossings resulting from pairing phrases in place

Phrase Type Alignment

In addition to token-based alignments, we also abstract sentences to a sequence of shallow parse labels and again compute an alignment and unaligned tokens between sentences. The above example would have the same shallow parse representation for both premise and hypothesis, so they would have a perfect alignment with no remaining unaligned tokens:

[NP A brown and white dog] [VP is running] [PP through] [NP the tall grass] → NP VP PP NP
[NP A brown and white dog] [VP is moving] [PP through] [NP the wild grass] → NP VP PP NP

From the shallow parse label sequence for the premise and hypothesis, we compute the longest label matching subsequence and compute the following features:

- Lengths of premise and hypothesis label sequences
- Length of longest matching label subsequence
- Ratio of the length of longest matching label subsequence to the length of the label sequence for both the premise and the hypothesis
- Numbers of unaligned labels in premise and hypothesis
- Ratio of the number of unaligned labels to the length of the label sequence for both the premise and the hypothesis

4.2.6 Lexical relations

We include features to count occurrences of hypernyms, synonyms, and antonyms between the premise and the hypothesis. These relations are more precise than the phrase similarity features, as a highly similar phrase pair may contain synonyms or antonyms, but they have lower coverage than distributional similarity. For these features, we consider primarily the pairs of highly similar phrases that we identified by aligning phrases with distributional similarity (we also consider words that could not be paired with distributional similarity due to out-of-vocabulary issues).

Hypernyms

The hypernym features count the number of hypernyms in the similar phrases between the premise and the hypothesis: one feature counts the words in the premise phrases that have a hypernym in the hypothesis phrases, and another feature counts the words in the hypothesis that have a hypernym in the premise. We identify hypernyms using WordNet, assuming that all WordNet senses for a given word are valid.

To deal with out-of-vocabulary issues stemming from our distributional vector representation, which is only computed over FLICKR30K, we additionally count hypernym-hyponym pairs in any remaining phrases that could not be paired using distributional similarity.

Synonyms

Synonym features count the number of synonym pairs in the same candidate phrases. Synonyms are words that share a WordNet synset, assuming that all WordNet senses are valid for any word.

Antonyms

We first count as antonyms any occurrences of the following patterns between premise and hypothesis:

- *X–not X*
- *X–no X*
- *X–no* `HeadNoun`*(X)* (e.g. *blue hat–no hat*)

- *X–no* `Hypernym(X)` (e.g. *poodle–no dog*)
- *X–no* `Synonym(X)` (e.g. *kid–no child*).

Secondly, we want to identify antonyms in similar phrase pairs. We consider a broader definition of antonyms than usual. In addition to standard antonyms like *crowded* street vs. *empty street*, we also want to identify word pairs that are mutually exclusive in the context of textual entailment, like *man* and *woman* or *bike* and *car*.

To do this, we identify similar phrases that occur at least twice in NEUTRAL or CONTRADICTION sentence pairs in the training data and do not occur in any ENTAILMENT sentence pairs. Commonly matched chunks in NEUTRAL or CONTRADICTION sentence pairs include *sit–stand*, *boy–girl*, and *cat–dog*. We use these pairs to define an antonym dictionary and count these occurrences in the antonym feature.

The information captured by these antonym pairs is similar to the knowledge that we aim to capture with the denotation graph. Phrases with low denotational similarity should not be used to describe the same scene. Similarly, these antonym pairs indicate that two phrases, while similar, describe two distinct, non-overlapping scenarios.

4.3 EXPERIMENTS

For semantic relatedness, we implement a log-linear ridge regression model [69] using Weka [70]. For textual entailment, we use a logistic regression classifier implemented with MALLET [59]. For both models, we use the default parameters and the same set of features.

4.3.1 Preprocessing

We lemmatize all sentences with Stanford CoreNLP[1] and extract shallow parses for the alignment features using the Illinois Chunker [71]. For features that require stopword removal, we use the NLTK[2] English stopword list (127 words). We remove negation words (*no*, *not*, and *nor*) from the stopword list since their presence is informative for this dataset.

4.3.2 Results

Table 4.3 shows the semantic relatedness results of our model [35] on the test data compared to the top five systems as well as the task baseline. Table 4.4 shows our textual

[1] `https://stanfordnlp.github.io/CoreNLP/`
[2] `https://www.nltk.org/`

	Pearson r
BASELINE	0.627
Lai and Hockenmaier [35]	0.799
Jimenez et al. [26]	0.804
Bjerva et al. [23]	0.827
StanfordNLP	0.827
Zhao et al. [31]	**0.828**

Table 4.3: The top 5 SemEval-2014 results for semantic relatedness (Pearson correlation on test data). Our model was fifth out of 17 semantic relatedness systems.

	Accuracy
BASELINE	56.2
Jimenez et al. [26]	83.1
Zhao et al. [31]	83.6
Lai and Hockenmaier [35]	**84.6**
Beltagy et al. [32]	85.1
Yin et al. [33]	86.2
Yin and Schütze [34]	**87.1**

Table 4.4: The top 3 SemEval-2014 results for textual entailment (accuracy on test data) compared to recent state-of-the-art results. Our model was first out of 18 systems in the shared task.

entailment results compared to the task baseline, the other top systems from the shared task, and more recent state of the art advances in textual entailment on SICK. Overall, our model performed quite well, ranking fifth out of 17 systems for semantic relatedness and first out of 18 systems for textual entailment. Like our system, most of the other top performing systems used a combination of features, often including a compositional vector representation, and limited their use of external knowledge sources to WordNet and paraphrase corpora.

In the rest of this section, we present an ablation study of our different features to determine their various contributions.

Model	Accuracy			
	Overall	Entailment	Neutral	Contradiction
BASELINE	56.8	44.8	77.3	0.0
All features	84.2	83.3	86.5	77.0
Word overlap	65.0	63.8	82.9	0.0

Feature	Feat. removal acc. Δ				Feat. addition acc Δ			
	Overall	E	N	C	Overall	E	N	C
Word overlap	−0.2	0.7	−0.4	−1.4	–	–	–	–
Negation	**−2.6**	**−6.3**	0.3	**−6.8**	10.2	2.8	−3.2	**74.3**
DenSim	−1.0	−2.8	−1.1	2.7	9.4	3.5	0.7	52.7
DistSim	−0.4	0.0	−0.4	−1.4	6.8	−3.4	3.6	37.8
Den+Dist	−1.6	−4.2	−1.4	2.7	12.0	4.9	2.5	60.8
Alignment	0.6	0.7	0.7	0.0	5.4	−13.2	5.0	41.8
Unaligned	−0.6	0.0	−0.7	−1.4	**10.8**	2.8	**7.5**	37.8
Synonyms	0.2	1.4	0.0	−1.4	0.2	1.4	−0.7	0.0
Hypernyms	−1.4	0.0	**−2.2**	−1.4	6.0	4.2	1.1	0.0
Antonyms	0.0	−2.8	1.7	−1.4	6.0	**18.8**	0.7	0.0

Table 4.5: Feature ablation results for textual entailment on development data

4.4 ANALYSIS

4.4.1 Feature Ablation

To determine which features contribute the most to the overall performance of our model, we performed an ablation study. We train the model on the training data and evaluate on the validation data with a different combination of features each time. The results of these experiments are in Table 4.5. The top half of the table contains the accuracy of our full model with all features on the validation data as well as its accuracy on each label. We compare this to the same classifier trained with only our single word overlap feature.

The bottom half of the table contains the ablation results for each feature group. The left side shows the change in accuracy from the full model when we remove the specified feature. Larger negative values indicate that the feature is more important to the overall performance of the model. The right side shows the difference in accuracy when we add the specified feature to word overlap. Larger positive values indicate that the feature group produces larger gains on top of the word overlap feature.

Negation is clearly is the most important feature for our overall textual entailment performance, resulting in the largest drop in accuracy when it is removed from the model (a

decrease of 2.6 points from 84.2%). Hypernym features are the second most important group. From the right side of the table, we can see that as we expected, adding hypernym features increases accuracy on ENTAILMENT by 4.2 points. However, there is some overlap with other features: from the left side, we see that removing hypernym features from the full model primarily decreases the accuracy on NEUTRAL and CONTRADICTION. We hypothesize that for NEUTRAL and CONTRADICTION sentence pairs, the absence of a hypernym pair in conjunction with other feature values may indicate the correct category.

Denotational constituent similarity features contribute noticeably to the overall performance, and are one of the feature groups to add the most improvement (9.4 points) over the word overlap feature. Our ablation analysis shows that denotational similarity and distributional similarity have different strengths: denotational features contribute more to textual entailment than distributional features, but we saw the opposite effect for semantic relatedness [see 35, Table 3]. This follows our intuition that denotational similarity features are particularly suited for entailment-type tasks, while distributional similarity captures the more general concept of topical similarity. The two types of similarity capture complementary information: removing both distributional and denotational constituent similarity features results in lower performance on both textual entailment and semantic relatedness than removing either feature individually. Notably, denotational similarity performed well on textual entailment even though we used PMI which is not a directional similarity metric.

4.4.2 Coverage of Constituent-Based Features

Since the denotation graph was not constructed on the SICK data, there may be some sentence constituents do not exist in the graph with sufficient frequency (fewer than 10 images in FLICKR30K) or at all. In that case, the constituent will have no denotational similarity to any other constituent. For example, a phrase like *exhausted man* which we have not seen before in the training data will have a denotational similarity score of 0.0 with any other phrase. As a result, denotational similarity features may have poor coverage. Since our distributional features are compositional, they have much better coverage: we can use the vector for *man* and take the cosine similarity of $\langle$*man*, *person*$\rangle$ as an approximation for the similarity of $\langle$*exhausted man*, *person*$\rangle$

In Table 4.6, we quantify the coverage of the denotational and distributional features across different constituent types. The coverage is the percent of constituent features with a non-zero value (we ignore constituents where there was no comparison to be made, e.g. if we did not identify a direct object in one of the sentences). We only show the coverage for the LEAFSIM constituent features, which compare the constituent leaf nodes of the same type

	% of instances covered				
Features	subj	VP	verb	dobj	NP
Denotation	65.8	30.1	41.9	70.6	80.3
Distributional	97.3	96.7	92.3	91.2	97.1

Table 4.6: The percentage of constituent pairs that have nonzero feature values for LeafSim constituent comparisons. Distributional features have higher coverage than denotational features

between sentences. The MixSim and AncSim features alleviate some coverage issues by considering parent and grandparent nodes, but the coverage of those features shows similar patterns.

Unsurprisingly, the coverage of the distributional representations is always better than the coverage of the denotational representation. However, despite incomplete coverage, denotational similarity features still noticeably contribute to the overall performance of the model, as we observed in the ablation experiment. Improving the coverage of denotational similarity could further improve its contributions to textual entailment.

4.4.3 Strengths and Weaknesses

In this section, we discuss specific errors that our model makes. We provide examples of difficult and ambiguous cases in the dataset as well as examples of sentence pairs that our model correctly classifies. Table 4.7 contains the sentence pairs under discussion.

Some of our model's misclassifications are actually sentence pairs that are inconsistently labeled. Example 1, where the premise explicitly negates the hypothesis, is labeled as NEUTRAL, while Example 2, which contains the antonym pair *standing* vs. *running*, is labeled as CONTRADICTION. Our model reverses the labels, making two incorrect predictions. The inconsistent annotation of sentences involving negation makes it difficult to train a model to distinguish between CONTRADICTION and NEUTRAL sentence pairs. We discuss this annotation issue further in Section 4.5.2.

Our model also has trouble handling multiword paraphrases. Correctly labeling Example 3 requires the knowledge that a *green ball* and a *green colored ball* are equivalent, information that our model does not have. An even more difficult case is Example 4, which requires a deeper understanding of what *a snowboarder doing a flip* involves: that there is a person on a board who is jumping in the air.

Our model also has difficulty with pairs where the key semantic difference between the two sentences is in the prepositional phrase. Our model incorrectly predicts the labels for

	Premise	Hypothesis	Gold	Pred
1	A man with no hat is sitting on the ground	A man with a backwards hat is sitting on the ground.	N	C
2	A black and white dog with a large branch is standing in the field.	A black and white dog with a large branch is running in the field.	C	N
3	A large green ball is hitting a potato.	A large green colored ball is hitting a potato.	E	C
4	The snowboarder is doing a flip over a mound of snow.	Somebody is jumping in the air on a board.	E	N
5	Five children are standing in front of a wooden hut.	Five children are standing in a wooden hut.	N	E
6	A woman dressed in elegant clothing is inside a crowd of people and is looking up.	A woman dressed in elegant clothing is inside a crowd of people and is looking down.	C	E
7	A few men in a competition are running outside.	A few men in a competition are running indoors.	C	N
8	Some people and vehicles are on a crowded street.	Some people and vehicles are on a almost empty street.	C	E
9	The milk is being drunk by a cat.	The cat is drinking some milk.	E	E
10	A man is doing a trick on a skateboard.	A person is doing a trick on a skateboard.	E	E
11	A hurdle is being leapt by a horse that has a rider on its back.	A horse and its rider are leaping over a barrier.	E	E
12	There is no woman using an eye pencil and applying eye liner to her eyelid.	A woman is applying cosmetics to her eyelid.	C	C

Table 4.7: Sentence pair examples with gold and predicted textual entailment labels from SICK development data.

both Example 5 and Example 6, where adding or altering a single word in the prepositional phrase changes the relationship from ENTAILMENT to NEUTRAL (removing *front*) or CONTRADICTION (*up* $\rightarrow$ *down*).

Finally, although the antonym feature does help our model identify some CONTRADICTION pairs, there are still several that we miss, such as Example 7 (*outside* $\neq$ *indoors*) and Example 8 (*crowded* $\neq$ *almost empty*).

Our model generally correctly classifies passive transformations such as Example 9. Although the typed constituent features rely on comparable syntactic roles between the two sentences, the untyped features allow the comparison of different constituent types. In addition, the alignment features attempt to match up the unaligned words in the sentence pair regardless of word order, which allows active-passive sentence pairs to achieve higher

Meaning-Preserving Transformations	
Active ↔ Passive	A man is driving a car. → A car is being driven by a man.
Replace words with (near) synonyms	A young boy is jumping into water → A young kid is jumping into water
Add modifiers that do not radically alter meaning	A deer is jumping a fence → A wild deer is jumping a fence
Meaning-Altering Transformations	
Insert negation	The boy is playing the piano → The boy is not playing the piano
Replace word with a semantic opposite	The girl is spraying the plants with water → The boy is spraying the plants with water
Scramble words	The turtle is following the fish → The fish is following the turtle

Table 4.8: Examples of the two types of transformations, meaning-preserving and meaning-altering, that were used to generate SICK sentences.

similarity scores.

Expanding our model's vocabulary using WordNet synonyms and hypernyms helps us most on sentence pairs that have high word overlap but one key word differs, such as Example 10 or Example 11. Our model correctly classifies both pairs as ENTAILMENT, because it knows that *person* is a hypernym of *man* and *barrier* is a hypernym of *hurdle*. Our model also correctly labels Example 12 using the information that *cosmetics* is a hypernym of *(eye) pencil*.

4.5 NOTABLE DATASET PHENOMENA

The sentences in SICK were generated by simplifying the original descriptive captions and then applying a set of transformation rules. The rules included meaning-preserving rules like transformating active sentences to passive sentences, as well as meaning altering rules like inserting negation. Table 4.8 contains some of the transformation rules and corresponding example sentences.

4.5.1 Phenomena Resulting from Rule-Based Sentence Transformations

Since the hypotheses in SICK are generated by applying rule-based transformations to the premises, they avoid some of the bias that results from asking human annotators to write

a hypothesis sentence given a premise and a particular entailment label [12]. However, the way in which the SICK transformations were applied resulted in other biases.

Two types of transformations were used to generate new sentences from the normalized sentences: meaning-preserving transformations and meaning-altering transformations. The resulting transformed sentences are paired almost exclusively with the original sentence: only 12% of the sentence pairs result from pairing random, unrelated sentences [see 8, Table 5]. As a result, the meaning-preserving transformations are strongly associated with sentence pairs that are ultimately labeled as ENTAILMENT, meaning-altering transformations are associated with CONTRADICTION, and word-scrambling transformations are strongly associated with NEUTRAL pairs. Any model that can identify the patterns in these transformations can fairly easily label the same-set sentence pairs that make up almost half of the dataset.

The most obvious example of this is the negation insertion transformation. As one of the meaning-altering transformations, it occurs primarily with CONTRADICTION pairs and occasionally NEUTRAL pairs. It never appears in a sentence pair that has been labeled ENTAILMENT. As a result, a classifier trained with a single binary negation feature like ours has over 60% overall accuracy on the development data. With just two features – binary negation and word overlap – a simple model achieves over 75% accuracy.

Similarly, sentences produced via meaning-preserving transformations (e.g. turning an active sentence into a passive sentence) never indicate CONTRADICTION. Based on this observation, it is likely that a model with a simple syntax-based feature similar to the binary negation feature would also have surprisingly high accuracy.

When building a textual entailment dataset, it is important to keep in mind the extent to which the data can be "hacked" by features like these which are based on lexical or syntactic patterns and do not truly capture anything about the meaning of a sentence or what entailment really is. All datasets are subject to bias of one kind or another, but building challenging datasets requires a thorough exploration of different kinds of baselines to minimize these occurrences.

4.5.2 Sentences Containing Explicit Negation

The sentence pairs that contain explicit negation are a unique feature of the SICK dataset, as previous textual entailment datasets did not contain a significant number of these cases (the Framework for Computational Semantics (FRACAS) entailment test suite [72] contains negated plurals, but not negation of verb phrases as SICK does). However, annotators were not provided with specific instructions as to how to label these cases of negation. As a result, we observe that these pairs are inconsistently labeled across the entire dataset.

For example, the following example is labeled NEUTRAL despite the underlined explicit negation:

PREMISE: A man with no hat is sitting on the ground.
HYPOTHESIS: A man with a backwards hat is sitting on the ground.

⇒ NEUTRAL

By contrast, the next example is labeled CONTRADICTION due to the opposed meaning of the verbs.

PREMISE: A black and white dog with a large branch is standing in the field.
HYPOTHESIS: A black and white dog with a large branch is running in the field.

⇒ CONTRADICTION

Marelli et al. [3] do mention that annotators treated potentially contradictory sentence pairs where subjects are accompanied by indefinite determiners differently from sentence pairs whose subjects are marked by definite determiners. However, in the above example, both sentence pairs contain subjects with indefinite determiners. Since the majority (64 of 74 pairs in the development data) of CONTRADICTION sentence pairs involve explicit negation, this distinction is critical to understanding what constitutes CONTRADICTION in SICK.

In making their own textual entailment dataset, SNLI, Bowman et al. [4] observed that the distinction between CONTRADICTION and NEUTRAL hinges on whether the annotator assumes that the premise and hypothesis refer to the same single event (and same single entity). Bowman et al. attempt to address the issue by informing the annotators that the premise and hypothesis sentences refer to the same grounded scene (an image which the annotators are not shown). This does not completely solve all questions about entity coreference between premise and hypothesis, but it would resolve many of the negated sentence pairs in SICK to be CONTRADICTION. Ideally, future annotation efforts for textual entailment datasets should endeavor to disambiguate these difficult cases as much as possible to produce consistent annotations.

4.6 CONCLUSION

This chapter presented a detailed description of the model that we submitted to the SemEval 2014 shared task for textual entailment and semantic relatedness. Our model included denotational similarity features based on decomposing new sentences into constituent phrases that exist in the pre-computed FLICKR30K denotation graph. We presented ablation experiments analyzing the contributions of our features. In particular, we observe

that denotational similarities and distributional similarities computed over the same text are complementary: denotational similarity, even symmetric PMI, is more effective for textual entailment, while distributional similarity contributes more to the semantic relatedness task.

Furthermore, we also analyze the construction of the SICK dataset. While the rule-generated hypotheses may lack some biases of human-written hypotheses, the rules used to generate the premise and hypothesis sentences are extremely important. Uneven application of these rules results in a dataset where a few simple features can achieve high accuracy without true language understanding. We take these lessons into account when we construct our own entailment dataset in Chapter 6.

Now that we have demonstrated that denotational similarity can be straightforwardly applied to existing semantic tasks as an explicit feature, we next extend it to vector representations. In this chapter, we applied denotational similarity to phrases that we previously observed in the Flickr30K denotation graph. In the next chapter, we present an embedding model to alleviate this issue.

CHAPTER 5: A STRUCTURED EMBEDDING SPACE FOR DENOTATIONAL PROBABILITIES

We have shown that denotational similarities are useful features for semantic tasks like textual entailment and semantic relatedness. In this chapter, we propose a framework that captures the denotational probabilities of words and phrases by embedding them in a structured vector space that captures denotational set relationships. We then present a method to induce such an embedding from a dataset of denotational probabilities. We show that our compositional model successfully produces this representation for unseen phrases, extending the application of denotational probabilities to textual entailment datasets like SNLI.

5.1 AN ORDER EMBEDDING FOR PROBABILITIES

5.1.1 Entailment Embeddings

Dense vectors, or embeddings, have become a common representation for words. Traditional linear models often consider lexical features as indicator functions, where each feature indicates, for example, that a word was present in the text. However, this one-hot feature representation loses a lot of information by assuming that each token corresponds to a unique vector dimension: under this assumption, *cat* and *cats* are two distinct, unrelated words and there is no opportunity to share information between them. A dense word vector, by comparison, represents each word as a dense vector in some d dimensional space. Rather than a one-hot vector where a single dimension has a value of 1 and all other dimensions have a value of 0, a dense word representation can have non-zero values in all dimensions. These values may come from a more direct feature representation, like counting the co-occurrences between words in a sliding window over an entire corpus, or they may be the result of training a neural network model for some other task starting from randomly initialized word representations.

Dense vector representations can improve generalization in our models. For example, we may have observed the word *child* frequently during training, but only saw *toddler* a few times. A model that uses a one-hot representation that compares words based on the surface string identity will not have much information about *toddler*. However, if the dense vectors

Work in this chapter was first published in A. Lai and J. Hockenmaier (2017), "Learning to Predict Denotational Probabilities for Modeling Entailment," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 721–739 [73].

for *child* and *toddler* are similar, then our model may be able to leverage features of *child* to share with *toddler*. This does assume that we observed *toddler* frequently enough in training to build a vector representation that is close to *child*, or that we use pre-trained word embeddings based on a separate, large corpus where this is true.

Pre-trained embeddings are now a common starting point for many NLP tasks where the labeled corpus size is much smaller than the corpus used to train the embeddings. Lexical embeddings can be combined to produce dense representations for phrases and sentences that are useful to the particular task at hand. Initially, these combination functions were simple applications of vector addition or element-wise multiplication, but the current accepted approach is to produce a sentence vector by feeding a sequence of word vectors into a neural sequence model, such as an RNN. We take this approach in this chapter, using pre-trained embeddings to train a neural sequence model that produces phrase embeddings that reflect the information captured in the denotation graph. The resulting model can produce embeddings for phrases that we have not seen before.

Several related works have explored different approaches to learning vector space representations that express entailment more directly. Kruszewski et al. [74] learn a mapping from an existing distributional vector representation to a structured Boolean vector representation that expresses entailment as feature inclusion. They evaluate the resulting representation on lexical entailment tasks and on sentence entailment in SICK, but they restrict SICK to a binary task and their sentence vectors result from simple composition functions (e.g. addition) over their word representations. Henderson and Popa [75] learn a mapping from an existing distributional vector representation to an entailment-based vector representation that expresses whether information is known or unknown. However, they only evaluate on lexical semantic tasks such as hyponymy detection.

Other approaches explore the idea that it may be more appropriate to represent a word as a region in space instead of a single point. Erk [76] presents a word vector representation in which the hyponyms of a word are mapped to vectors that exist within the boundaries of that word vector's region. Vilnis and McCallum [77] use Gaussian functions to map a word to a density over a latent space. Both papers evaluate their models only on lexical relationships.

Most relevant to our model is the order embedding model of Vendrov et al. [78], who observed that the lexical hypernym relationship, the textual entailment relationship between sentences, and the image-caption relationship can all be seen as part of a larger partial order over language and images. In this partial order, a sentence $s =$ *"A woman is walking a dog"* that describes some image i is an abstraction of that image. Similarly, the sentence $s' =$ *"A person is walking a dog"* is entailed by s; s' is an abstraction of both s and i, as are the

phrase *woman* and its hypernym *person*.

From this definition, Vendrov et al. then define an *order embedding* for phrases and images such that the vector $\mathbf{y}$ corresponding to the phrase/image y is smaller than the vector $\mathbf{x}$, i.e. $\mathbf{y} \preceq \mathbf{x}$, for phrases/images x that are entailed by y, where $\preceq$ corresponds to the reversed product order on R_+^N ($\mathbf{y} \preceq \mathbf{x} \Leftrightarrow y_i \geq x_i \forall i$). The authors use their model to predict entailment labels between pairs of sentences, but it is only capable of making binary entailment decisions. They also evaluate their model on hypernym relations in WordNet and caption-image retrieval [49, 79].

The model we present in this chapter is based on this idea that directional phrase relationships can be encoded in an embedding space. Our model only handles text representations, not images, but we extend the binary ordered relationship to express the probability, from 0 to 1, that one phrase implies another.

5.1.2 Denotational Conditional Probability

In the previous chapter, we used pointwise mutual information as the denotational feature between phrases. For this model, however, we focus instead on denotational conditional probability (Equation 3.4), as it is directional and intended to capture entailment-like relations. In an ideal representation, if the premise p entails the hypothesis h, then the denotational conditional probability $P_{[\![}(h|p)$ should be 1 (or close to 1). Conversely, if h contradicts p, then $P_{[\![}(h|p)$ should be close to 0. The denotation graph is constructed to express some entailment relations directly: if x is an ancestor of y in the graph, then y entails x and $P_{[\![}(x|y) = 1$. We therefore stipulate that learning to predict the denotational conditional probability $P_{[\![}(h|p)$ would be helpful in predicting textual entailment.

5.1.3 Defining a Structured Embedding Space

We generalize Vendrov et al.'s binary entailment order embedding to an embedding space that expresses the denotational probability that phrase x is true given phrase y. Denotational probability represents a phrase as a set of scenarios accurately described by that phrase, and the conditional probability of phrase x given phrase y is the set intersection, the number of overlapping scenarios, of x and y. In translating this representation to a vector space, we can envision each vector, each point in the space, as corresponding to a particular possible scenario in the set of all possible scenarios. Then $P_{[\![}(x)$ would correspond to a set of vectors, possibly encompassing a region in the vector space. If $P_{[\![}(x)$ and $P_{[\![}(y)$ each correspond to a region of points in the space, then we can take the overlap in their regions as the

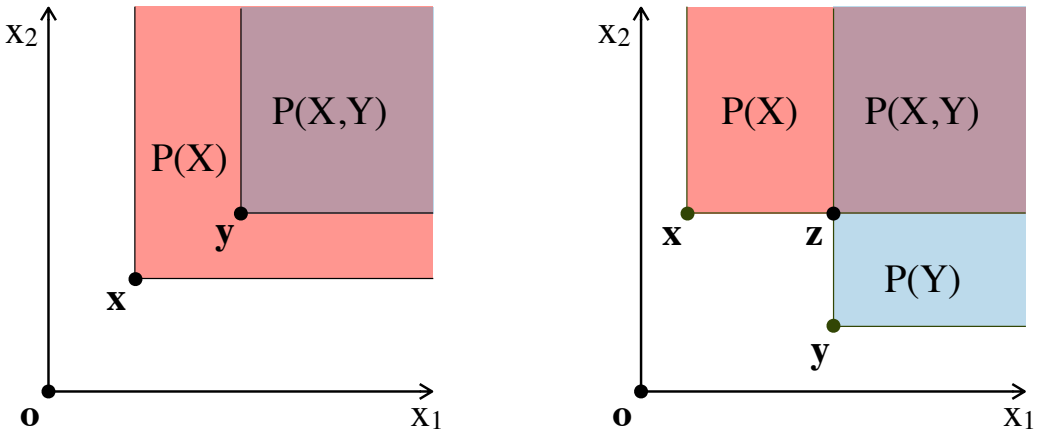


Figure 5.1: An embedding space that expresses the individual probability of events X and Y and the joint probability $P(X, Y)$.

set intersection corresponding to $P_{\amalg}(x, y)$ from which we can compute the denotational conditional probability.

Concretely, we want to map a phrase x to a vector $\mathbf{x}$ that corresponds to a region in the embedding space that is proportional to $P_{\amalg}(x)$. We also want the overlap between any two vector regions $\mathbf{x}$ and $\mathbf{y}$ to correspond to the joint probability $P_{\amalg}(x, y)$, from which we can compute $P_{\amalg}(x|y)$.

Figure 5.1 illustrates the basic idea of this embedding space in two dimensions. Each phrase vector occupies a region proportional to the denotational probability of the phrase and corresponding to the set of points greater than the vector in each dimension. The denotation of the entire universe (all possible scenarios) corresponds to the entire region of the positive orthant, i.e. the origin vector. All other points in the space correspond to smaller regions within the positive orthant and thus probabilities less than 1.

Concretely, we learn a mapping from phrase x to an N-dimensional vector $\mathbf{x} \in \mathbb{R}_+^N$ such that $\mathbf{x} = (x_1, ..., x_N)$ defines the denotational probability of x as $P_{\amalg}(x) = \exp(-\sum_i x_i)$. The origin has probability $\exp(0) = 1$, while any other vector $\mathbf{x}$ such that $\exists_i x_i > 0$ has a denotational probability less than 1. When comparing a pair of vectors, if $\mathbf{x}$ is farther from the origin than $\mathbf{y}$, then phrase x has a smaller denotational probability than phrase y.

The joint probability $P_{\amalg}(x, y)$ in this embedding space is proportional to the size of the intersection of the regions of $\mathbf{x}$ and $\mathbf{y}$. We define this joint probability as the region that corresponds to the vector $\mathbf{z}$ that is the element-wise maximum of $\mathbf{x}$ and $\mathbf{y}$:

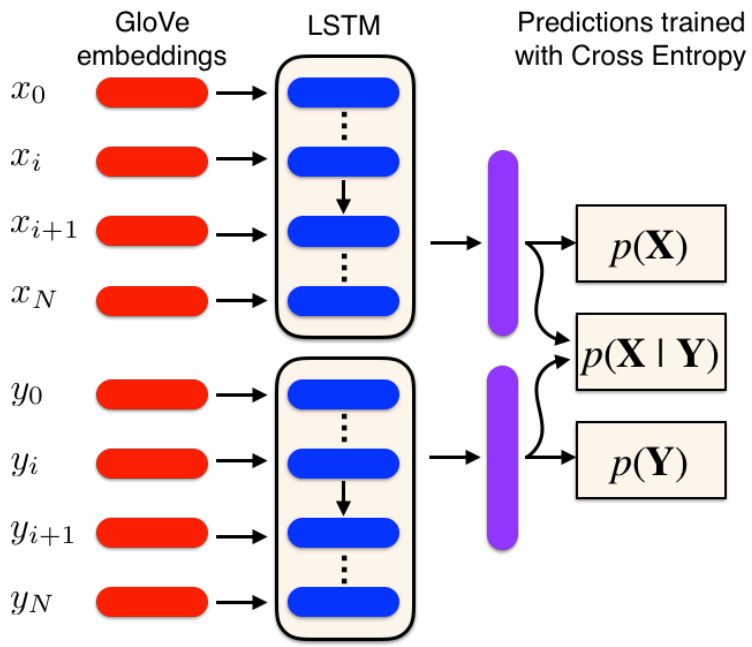


Figure 5.2: Our probability model DENEMBED. Each phrase is a sequence of word embeddings that is passed through an LSTM to produce a 512d vector representation for the premise and the hypothesis. Both vectors are used to compute the predicted conditional probability and calculate the loss.

$$\mathbf{z} = \max(x, y) : z_i = \max(x_i, y_i) \tag{5.1}$$

This allows us to compute the conditional probability $P_{\mathbb{I}}(x|y)$ as follows:

$$\begin{aligned} P_{\mathbb{I}}(x|y) &= \frac{P_{\mathbb{I}}(x, y)}{P_{\mathbb{I}}(y)} \\ &= \frac{\exp\left(-\sum_i z_i\right)}{\exp\left(-\sum_i y_i\right)} \\ &= \exp\left(\sum_i y_i - \sum_i z_i\right) \end{aligned} \tag{5.2}$$

5.2 TRAINING A DENOTATIONAL EMBEDDING MODEL

5.2.1 Model Architecture

We train a neural network model to predict phrase probabilities $P_{[\![]\!]}(x)$ and $P_{[\![]\!]}(y)$ and the conditional probability $P_{[\![]\!]}(x|y)$ for phrase pairs $\langle x, y \rangle$. The model, which we will call DENEMBED, consists of a single LSTM that produces a 512d phrase vector $\mathbf{x}$ or $\mathbf{y}$ from a sequence of pre-trained GloVe vectors (we use the 300d vectors trained on 840B tokens) corresponding to the words in phrase x or y.

We take $\mathbf{x}$ and $\mathbf{y}$, the LSTM output vectors at the final timestep, and sum their elements to compute the predicted denotational probabilities $\hat{P}_{[\![]\!]}(x)$ and $\hat{P}_{[\![]\!]}(y)$. From $\mathbf{x}$ and $\mathbf{y}$, we also compute the joint vector $\mathbf{z}$ (Equation 5.1), which we use to compute the predicted denotational conditional probability $\hat{P}_{[\![]\!]}(x|y)$ according to Equation 5.2. Figure 5.2 illustrates the structure of our model.

Our training data consists of ordered phrase pairs $\langle x, y \rangle$. For each pair, the loss is the sum of cross-entropy losses for three predicted probabilities, $\hat{P}_{[\![]\!]}(x)$, $\hat{P}_{[\![]\!]}(y)$, and $\hat{P}_{[\![]\!]}(x|y)$:

$$\begin{aligned} L = &- \Big[P_{[\![]\!]}(x) \log \hat{P}_{[\![]\!]}(x) + \big(1 - P_{[\![]\!]}(x)\big) \log \big(1 - \hat{P}_{[\![]\!]}(x)\big) \Big] \\ &- \Big[P_{[\![]\!]}(y) \log \hat{P}_{[\![]\!]}(y) + \big(1 - P_{[\![]\!]}(y)\big) \log \big(1 - \hat{P}_{[\![]\!]}(y)\big) \Big] \\ &- \Big[P_{[\![]\!]}(x|y) \log \hat{P}_{[\![]\!]}(x|y) + \big(1 - P_{[\![]\!]}(x|y)\big) \log \big(1 - \hat{P}_{[\![]\!]}(x|y)\big) \Big] \end{aligned} \tag{5.3}$$

5.2.2 Numerical Issues

In Section 5.1.3, we described the probability vectors $\mathbf{x}$ as being in the positive orthant. However, in order to prevent the gradients from becoming too small during training, we use log probabilities in our implementation. This means our vectors are actually in the negative orthant.

We apply two modifications to the LSTM output vectors in order to compute $\hat{P}_{[\![]\!]}(x)$, $\hat{P}_{[\![]\!]}(y)$, and $\hat{P}_{[\![]\!]}(x|y)$. First, to ensure that $\mathbf{x}$ is in $\mathbb{R}^N_-$, we clip the element values of the output vector so that $x_i \leq 0$. Second, when computing the log probability $\log \hat{P}_{[\![]\!]}(x)$ from the phrase vector $\mathbf{x}$, we clip the sum of the elements of $\mathbf{x}$ to the range $(\log(10^{-10}), -0.0001)$ in order to avoid errors caused by passing $\log(0)$ values to the loss function.

The conditional log probability is simply $\log \hat{P}_{[\![]\!]}(x|y) = \log \hat{P}_{[\![]\!]}(x, y) - \log \hat{P}_{[\![]\!]}(y)$, where $\log \hat{P}_{[\![]\!]}(x, y)$ is now computed over the element-wise minimum of $\mathbf{x}$ and $\mathbf{y}$:

$$\log \hat{P}_{[\![]\!]}(x, y) = \sum_i \min(x_i, y_i) \tag{5.4}$$

The element-wise minimum is a standard pooling operation (min pooling instead of the more common max pooling). Note that if $x_i > y_i$, neither x_i nor y_i is updated with respect to the $\hat{P}_{[\![]\!]}(x|y)$ loss. Both x_i and y_i will always be updated with respect to the $\hat{P}_{[\![]\!]}(x)$ and $\hat{P}_{[\![]\!]}(y)$ components of the loss.

5.2.3 Denotational Phrase Data

We now define the DENPHRASE dataset[1] of denotational phrase pairs that we use to train DENEMBED. We start by identifying all phrase pairs that occur frequently enough in the denotation graph that we can rely on their individual and conditional probability values. Then we sample from this pool of phrase pairs to create the test and development data such that some phrases are unique to the test or development data.

From the training split of the FLICKR30K denotation graph, we identify all phrase pairs $\langle x, y \rangle$ that fit one of the following conditions:

- $P_{[\![]\!]}(x|y) > 0$ *s.t.* $|x| \geq 10, |y| \geq 10$
  (45 million pairs that have at least one image in common and where each phrase occurs with at least 10 images)

- $P_{[\![]\!]}(x|y) = 0$ *s.t.* $N \times P_{[\![]\!]}(x)P_{[\![]\!]}(y) \geq N^{-1}$
  (2 million pairs that have no common images where we would have expected at least one (given independence and the total number of images N))

- $P_{[\![]\!]}(x|y) = 1$ *s.t.* x is an ancestor of y, $|x| \geq 10, |y| \geq 10$
  (3 million pairs that are ancestor-descendant pairs in the denotation graph)

Given this pool of phrase pairs, we take the set of phrases that occur in these pairs and sample 5% of them as *test phrases* to occur only in the test data and another 5% to occur only in the development data. The test data then contains all phrase pairs from the pool where at least one phrase is a test phrase. As a result, at least one phrase in each test pair will be unseen in the training or development data. The development data is constructed the same way from the development phrases.

Since all phrases in the denotation graph have been lemmatized, the phrases in DENPHRASE are lemmatized as well.

[1] Available at `https://github.com/aylai/EntailmentProbabilityEmbedding`.

	$P(x)$		$P(x\|y)$	
	KL	r	KL	r
Training data	0.0003	0.998	0.017	0.974
Full test data	0.001	0.979	0.031	0.949
Unseen pairs	0.002	0.837	0.048	0.920
Unseen words	0.016	0.906	0.127	0.696

Table 5.1: DENEMBED predicts the probability of unseen phrase pairs in DENPHRASE with high correlation to the gold probabilities.

5.3 CONDITIONAL PROBABILITY EVALUATION

We evaluate how well our embedding model can predict the denotational conditional probability of one phrase given another (or one sentence given another). We train DENEMBED on the 42 million phrase pairs in the DENPHRASE training data with batch size 512 for 10 epochs. We use the Adam optimizer [80] with default parameters, and a dropout rate of 0.5. These parameters were tuned on the development data, and we selected the model with the lowest KL divergence between gold and predicted conditional probabilities:

$$D_{KL}\left(P||Q\right) = P_{[\![]\!]}(x|y) \log \frac{P_{[\![]\!]}(x|y)}{\hat{P}_{[\![]\!]}(x|y)} + \big(1 - P_{[\![]\!]}(x|y)\big) \log \frac{1 - P_{[\![]\!]}(x|y)}{1 - \hat{P}_{[\![]\!]}(x|y)} \tag{5.5}$$

5.3.1 Unseen Phrase Pairs

We evaluate DENEMBED's denotational probability predictions for the 4.6 million phrase pairs in the DENPHRASE test split. Table 5.1 reports the mean KL divergence $D_{KL}(P||Q)$ between $P_{[\![]\!]}(x)$ and $P_{[\![]\!]}(x|y)$ compared to the predicted probabilities $\hat{P}_{[\![]\!]}(x)$ and $\hat{P}_{[\![]\!]}(x|y)$, and the Pearson correlation r, which expresses the correlation between the gold and predicted probabilities as a value between -1 (complete negative linear correlation) and 1 (complete positive linear correlation).

DENEMBED's predicted conditional probabilities are fairly accurate, reaching a correlation of $r = 0.949$ on the complete test data. On the subset of 123,000 test phrase pairs where both phrases are previously unseen, DENEMBED's predictions are almost as good, reaching $r = 0.920$. On the subset of 3,100 test phrase pairs where at least one word was unseen in training, the model's predictions are worse, only achieving a correlation of $r = 0.696$.

We specifically look at DENEMBED's predictions for phrase pairs where the gold $P_{[\![]\!]}(x|y)$ is either 0 or 1. The latter case reflects an important property of the denotation graph, since $P_{[\![]\!]}(x|y) = 1$ when x is an ancestor of y. More generally, we can interpret $P_{[\![]\!]}(h|p) = 1$

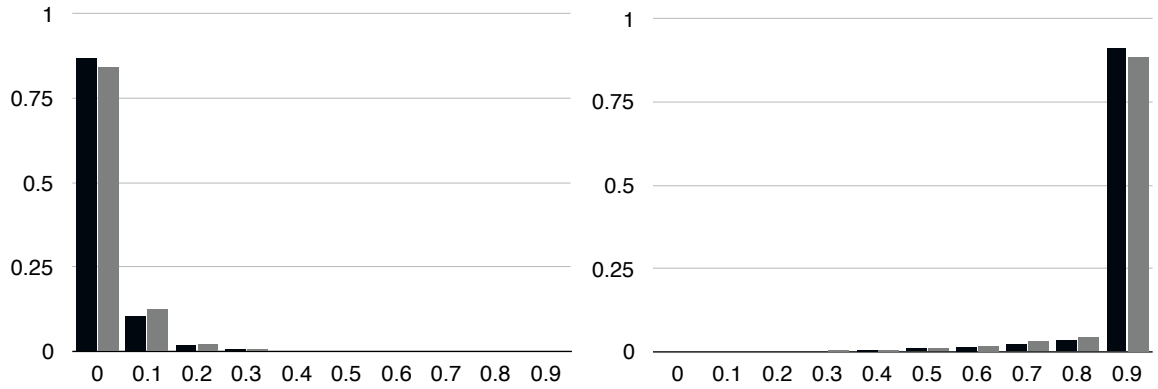


Figure 5.3: Predicted probabilities on a subset of DENPHRASE test data where $P_{\text{∏}}(x|y)$ is 0 (left) or 1 (right). Black is the full test data and gray is the subset of pairs where both phrases are unseen. Frequency is represented as a percentage of the pairs considered in each plot.

as a confident prediction of ENTAILMENT, and $P_{\text{∏}}(h|p) = 0$ as a confident prediction of CONTRADICTION. Figure 5.3 shows the distribution of predicted conditional probabilities for phrase pairs where gold $P_{\text{∏}}(h|p) = 0$ (left) and gold $P_{\text{∏}}(h|p) = 1$ (right). DENEMBED's predictions on unseen phrase pairs (gray bars) are nearly as accurate as its predictions on the full test data (black bars).

5.3.2 Longer Sentences

Up to this point, DENEMBED has only been trained on short phrases, since conditional probabilities in the denotation graph are only reliable for phrases that occur with multiple images (see Figure 5.4 for the distribution of phrase lengths in the DENPHRASE training data). To improve the model's performance on longer sentences, we use the 550,000 sentence pairs in the SNLI training data (which have a mean sentence length of 11 words) as additional data. We train a new model, which we will refer to as DENEMBED$_+$, on both DENPHRASE and SNLI (lemmatizing SNLI sentences to match DENPHRASE).

We augment the SNLI data with approximate gold denotational probabilities by assigning a probability $P_{\text{∏}}(S) = s/N$ to a sentence S that occurs s times in the N training sentences. We assign approximate gold conditional probabilities for each sentence pair $\langle p, h \rangle$ according to the entailment label: if p entails h, then $P(h|p) = 0.9$. If p contradicts h, then $P(h|p) = 0.001$. Otherwise, $P(h|p) = 0.5$.

Figure 5.5 shows the predicted probabilities on the SNLI test data when our embedding model is trained on different data distributions. The top row shows the predictions of DEN-

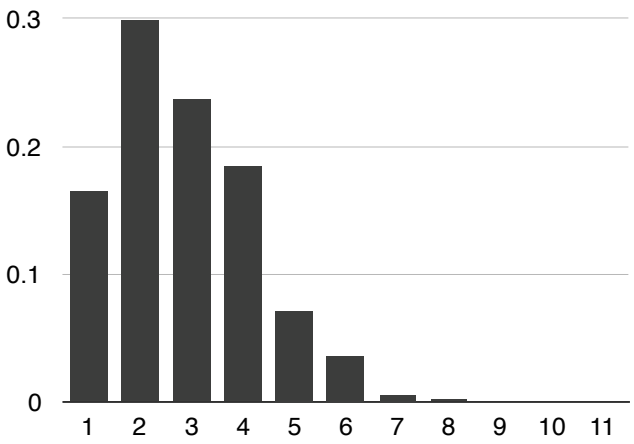


Figure 5.4: Distribution of phrase lengths as a fraction of the data size for DenPhrase training data.

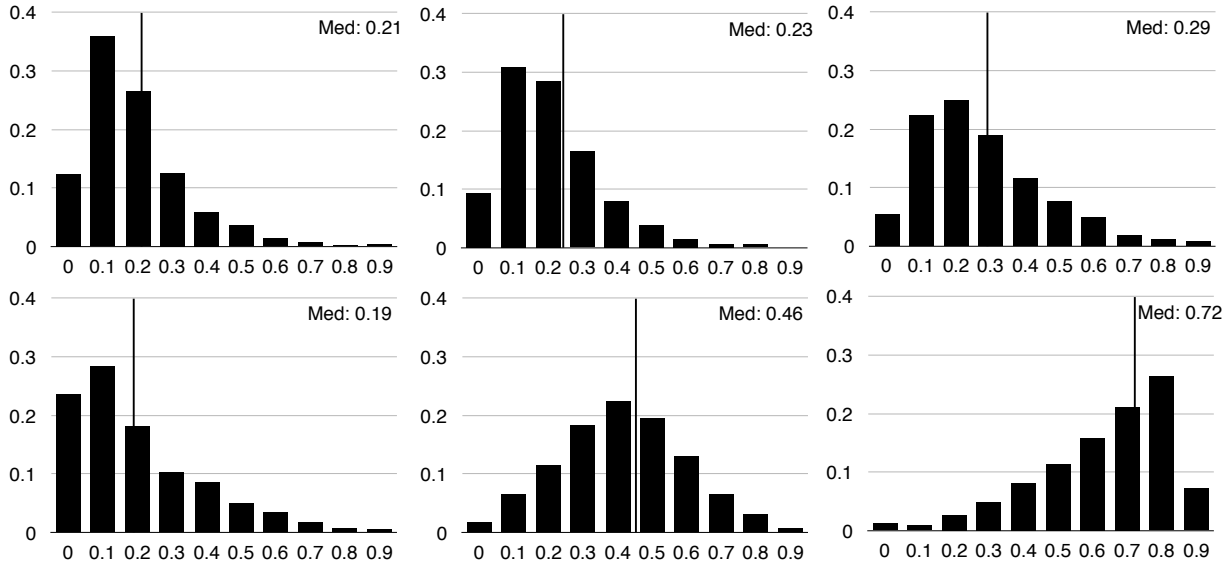


Figure 5.5: Predicted conditional probabilities $P(h|p)$ for SNLI sentence pairs (test) by entailment label, as a percentage of pairs with that label. Top: predictions from DenEmbed, which is trained only on DenPhrase. Bottom: predictions from DenEmbed$_+$, which is trained on both DenPhrase and SNLI.

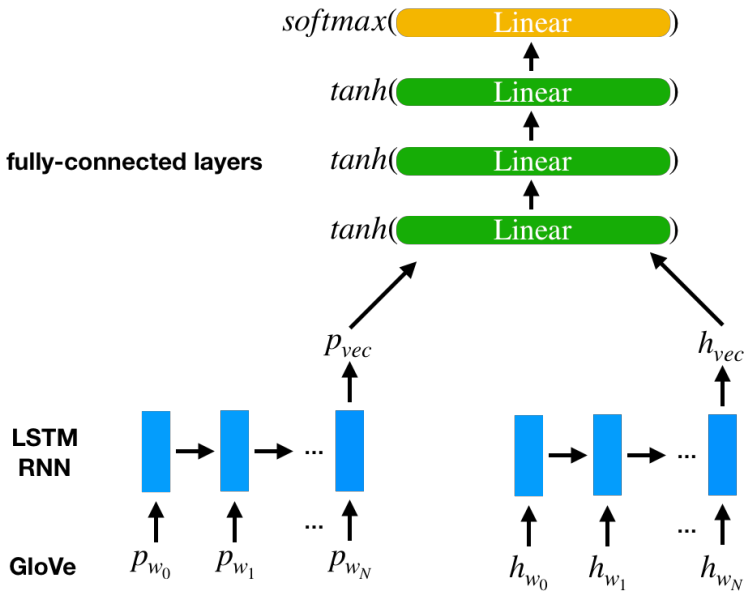


Figure 5.6: EntailLSTM: an entailment classifier based on an LSTM RNN that is applied to both the premise and the hypothesis.

Embed, which is trained only on DenPhrase. Given the training data, we did not expect these probabilities to align cleanly with the entailment labels, and indeed, the three distributions look fairly similar. However, the median probability of each class does increase from contradiction to neutral to entailment, confirming our intuition for how $P_{[\![}(h|p)$ should relate to textual entailment classes.

The bottom row shows that DenEmbed$_+$, which is trained on on both DenPhrase and SNLI with approximate conditional probabilities, has much improved probability predictions for longer sentences. DenEmbed$_+$'s predicted conditional probabilities align much more closely with the entailment class labels: entailment sentence pairs have high conditional probabilities (median 0.72), neutral pairs have mid-range probabilities (median 0.46), and contradiction pairs have probabilities approaching 0 (median 0.19).

5.4 TEXTUAL ENTAILMENT EVALUATION

We now evaluate the effectiveness of DenEmbed$_+$ for textual entailment, and demonstrate that these predicted probabilities are informative features for predicting entailment on both SICK and SNLI.

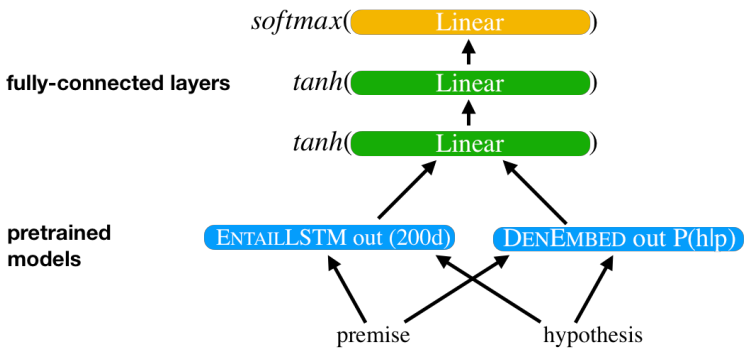


Figure 5.7: DenEnsemble: an entailment classifier that combines the outputs of pretrained EntailLSTM and DenEmbed models.

5.4.1 An Ensemble Model for Entailment

LSTM Entailment Model

We start with a basic LSTM entailment model [4], which we refer to as EntailLSTM (Figure 5.6). It consists of a single 100d LSTM RNN that takes sequences of GloVe vectors for both the premise and hypothesis, and produces 100d sentence vectors. The concatenated 200d sentence pair representation is passed through three 200d tanh layers and a softmax layer for three-class entailment classification. We train EntailLSTM with a batch size of 512 for 10 epochs. We use Adam with the default learning rate and a dropout keep rate of 0.85.

Combining the LSTM Entailment Model with Predicted Probabilities

We then combine the output of EntailLSTM (the result of the final tanh layer) with the output $\hat{P}_{[\![]}(h|p)$ of DenEmbed$_+$, and train a classifier over this conjoined feature representation. We refer to this classifier as DenEnsemble (Figure 5.7). It consists of two tanh layers that take the 201d feature vector as input, and a final softmax classification layer. Both EntailLSTM and DenEmbed have been pretrained (EntailLSTM on SNLI, and DenEmbed$_+$ on DenPhrase and SNLI), at which point we freeze their parameters and train DenEnsemble for 10 epochs on the relevant textual entailment data (SICK or SNLI) with a dropout keep rate of 0.5 and a batch size of 512.

Table 5.2 contains our results on SNLI and SICK. EntailLSTM achieves 77.2% accuracy on SNLI (the same accuracy reported by Bowman et al. [4]), whereas our combined model DenEnsemble improves the accuracy to 78.2% by including a single predicted denotational probability feature.

Model	Accuracy SNLI	SICK
EntailLSTM	77.2	81.5
DenEnsemble	**78.2**	**82.7**

Table 5.2: Entailment test accuracy on SNLI and SICK, comparing EntailLSTM, a standard LSTM classifier, to DenEnsemble, an ensemble method that adds DenEmbed$_+$'s predicted probability to the LSTM output.

We use a similar approach to evaluate our model on SICK. Since SICK is too small to train an LSTM, we combine the SICK and SNLI training data to pretrain EntailLSTM and then to train the combined model DenEnsemble. When we add the predicted conditional probability as a single feature for each SICK sentence pair, DenEnsemble's accuracy increases to 82.7%, compared to EntailLSTM's 81.5% accuracy. This approach outperforms the 80.0% accuracy achieved by Bowman et al. [4]'s transfer learning approach.

This experiment shows that the denotational embeddings contain information that is not learned by a standard neural entailment model. We have demonstrated that an LSTM entailment model can be improved by adding a single feature based on our predicted denotational probabilities.

5.4.2 Constituent-Based Features

In the previous experiment, we used a single predicted probability value $\hat{P}_{[\![]\!]}(h|p)$ to represent all the denotational similarity information between the premise and the hypothesis, replacing the large set of denotational similarity features we used in previous chapters. To evaluate whether we are losing information by reducing the number of denotational features, we apply the constituent decomposition method from Chapter 4 to sentences in SNLI, and use DenEmbed$_+$ to predict the conditional probability of each hypothesis-premise phrase pair, producing a total of 60 predicted probability features.

We now retrain DenEnsemble with EntailLSTM unchanged, but now with 60 predicted probabilities instead of 1. Table 5.3 contains our results. We consider constituent features with and without ancestor information from the graph. Without ancestor information, the constituent features perform worse than the single probability feature, achieving only 78.0% accuracy. However, including ancestor information pushes the accuracy to 78.5%, higher than the single probability feature. This indicates that there is some information, either involving ancestors or sub-phrase comparisons, that a single probability feature does not

Model	SNLI Accuracy
EntailLSTM	77.2
DenEnsemble	78.2
$\text{DenEnsemble}_{\text{const}}$	78.0
$\text{DenEnsemble}_{\text{anc}}$	**78.5**

Table 5.3: Entailment accuracy on SNLI test data. Appending multiple features based on ancestor constituent information results in higher accuracy compared to DenEnsemble, which only uses a single probability feature.

fully capture. However, DenEnsemble trained from a single predicted probability feature is still competitive with this larger set of features that requires more structural knowledge from the graph.

5.4.3 Binary Entailment Task

We also evaluate how well the predicted conditional probabilities $\hat{P}_{\mathbb{I}}(h|p)$ correspond to binary entailment labels. We follow Vendrov et al. [78] in transforming SNLI into a binary entailment task by relabeling all NEUTRAL and CONTRADICTION sentence pairs as NOT_ENTAILMENT and classifying them against ENTAILMENT pairs.

We use the predicted probability $\hat{P}_{\mathbb{I}}(h|p)$ for each SNLI sentence pair and tune a threshold $t \in \{0.0, 0.1 \ldots 0.9, 1.0\}$ on the development data. The best threshold produces 79.0% binary classification accuracy on the SNLI test data, significantly worse than Vendrov et al. [78]'s reported test accuracy of 88.6% from their order embedding model. Our result shows a strong correlation between the binary entailment label and the conditional probability of the hypothesis given the premise according to our probability model, but not sufficient to compete with order embeddings. However, as can be seen from the probability distributions in Figure 5.5, simply thresholding $\hat{P}_{\mathbb{I}}(h|p)$ does not produce unambiguous entailment classes. In the next section, we include some examples that demonstrate why it may not be a simple task to bin $\hat{P}_{\mathbb{I}}(h|p)$ into well-defined entailment classes.

	Premise	Hypothesis	$P_{[\![]\!]}$	$\hat{P}_{[\![]\!]}$
1	person walk on trail in woods	in forest	1.00	0.99
2	dark color clothing	clothing	1.00	0.85
3	group of person bike	group of person ride	0.86	0.78
4	adult sing while play instrument	adult play guitar	0.81	0.75
5	on busy street	sidewalk	0.35	0.73
6	person sit on bench outside	on park bench	0.42	0.42
7	child walk path	person wear green	0.07	0.31
8	tennis player hit ball	person swing	0.19	0.24
9	girl sleep	on pillow	0.06	0.23
10	man practice martial art	person kick person	0.13	0.26
11	person skateboard on ramp	man ride skateboard	0.18	0.20
12	busy intersection	city street	0.28	0.15
13	person dive into swim pool	person fly through air	0.10	0.11
14	sit at bench	adult read book	0.14	0.12
15	person leap into air	jump over obstacle	0.03	0.02
16	person talk on phone	man ride skateboard	0.01	0.02

Table 5.4: Gold and predicted conditional probabilities from the DenPhrase development data.

5.5 ANALYSIS

5.5.1 Model Predictions

In Section 5.3, we demonstrated that we can successfully predict denotational probabilities for phrases that we have not encountered during training and for longer sentences. In Section 5.4, we illustrated the utility of these probabilities by showing that a single feature based on our model's predicted denotational conditional probabilities improves the accuracy of an LSTM entailment model on SICK and SNLI by 1 percentage point or more. Although we did not evaluate the impact on more complex, recently proposed neural network models, this improvement is quite encouraging. We note in particular that we only have accurate denotational probabilities for the short phrases from the denotation graph (mostly six words or fewer), which have a limited vocabulary compared to the full SNLI data (there are 5263 word types in the denotation graph training data, while the lemmatized SNLI training data has a vocabulary of 31,739 word types). In light of these statistics, our modest gains are encouraging.

We examine examples of predicted conditional probabilities for both phrase and sentence pairs to identify the strengths and weaknesses of our denotational embedding model. Table 5.4 contains examples of DenEmbed's predictions for phrase pairs from the DenPhrase

Premise	Hypothesis	$P_{[\![]\!]}$	$\hat{P}_{[\![]\!]}$
skier on snowy hill	athlete	1.00	0.99
fan cheer	wear clothing and helmet	0.10	0.99
girl read book	adult read book	0.29	0.95
pitcher throw ball	mound	0.53	0.84
golf ball	athlete	0.53	0.66
wear gray jacket	person wear gray clothing	0.94	0.63
person point	man point	0.48	0.41
in front of computer	person look	0.36	0.21
person wear short and blue shirt	blue shirt	1.00	0.55

Table 5.5: Gold conditional probabilities for unseen pairs in the DenPhrase development data, compared to DenEmbed's predictions.

development data. DenEmbed correctly predicts high conditional probabilities for entailed phrase pairs even when there is no direct hypernym involved, as in example 3 (*biking* and *riding* are closely related but involve no synonym or hypernym relationship), and for closely related phrases that are not strictly entailing, as in example 4 (someone who is *singing and playing an instrument* is likely but not required to be playing a guitar). DenEmbed also predicts reasonable probabilities for events that frequently co-occur but are not required to do so, such as example 10: *practicing martial arts* sometimes involves *one person kicking another person*, but could also involve *punching* or *throwing* or *blocking* instead. In examples 13 and 14, DenEmbed predicts low but nonzero probabilities for occasionally co-occurring events (*diving* and *flying*, or *sitting on a bench* while *reading a book*), which are still more plausible and likely scenarios than the improbable co-occurrence in example 16 (a person *talking on the phone* while *riding a skateboard*). Table 5.5 demonstrates similar patterns for pairs where both phrases were unseen.

However, DenEmbed's predictions are not always so accurate. For example, in Table 5.4, examples 5 and 7 show overly high predicted probabilities for events that were either quite rare and unrelated (⟨*child walk path*, *person wear green*⟩) or for events that were only moderately connected (⟨*on busy street*, *sidewalk*⟩). Generally, DenEmbed is quite good at predicting $P_{[\![]\!]}(h|p) = 1.0$ for phrases involving hypernyms, but that is not true of example 2, where it predicts that the probability of *clothing* given *dark colored clothing* is only 0.85.

Table 5.6 contains examples of predicted conditional probabilities from DenEmbed$_+$ for SNLI development sentence pairs. In many cases, the label is obvious and so the probability of the hypothesis given the premise should also be easy to predict. Example 2 is a clear case of entailment that simply involves dropping words from the premise (*having drinks and smoking cigarettes*) to reach the hypothesis (*Two women are at a bar*). Example 1

		Premise	Hypothesis	$\hat{P}_{[\![\,]\!]}$
ENTAILMENT	1	A person rides his bicycle in the sand beside the ocean.	A person is on a beach.	0.88
	2	Two women having drinks and smoking cigarettes at the bar.	Two women are at a bar.	0.86
	3	A senior is waiting at the window of a restaurant that serves sandwiches.	A person waits to be served his food.	0.61
	4	A man with a shopping cart is studying the shelves in a supermarket aisle.	There is a man inside a supermarket.	0.47
	5	The two farmers are working on a piece of John Deere equipment.	John Deere equipment is being worked on by two farmers.	0.16
NEUTRAL	6	A group of young people with instruments are on stage.	People are playing music.	0.86
	7	Two doctors perform surgery on patient.	Two doctors are performing surgery on a man.	0.56
	8	Two young boys of opposing teams play football, while wearing full protection uniforms and helmets.	Boys scoring a touchdown.	0.30
	9	Two men on bicycles competing in a race.	Men are riding bicycles on the street.	0.24
CONTRADICTION	10	Two women having drinks and smoking cigarettes at the bar.	Three women are at a bar.	0.79
	11	A man in a black shirt is playing a guitar.	The man is wearing a blue shirt.	0.47
	12	An Asian woman sitting outside an outdoor market stall.	A woman sitting in an indoor market.	0.22
	13	A white dog with long hair jumps to catch a red and green toy.	A white dog with long hair is swimming underwater.	0.09
	14	Two women are embracing while holding to go packages.	The men are fighting outside a deli.	0.06

Table 5.6: DENEMBED$_+$'s predicted conditional probabilities for sentence pairs from the SNLI development data.

is a slightly more difficult case of ENTAILMENT where the hypothesis does not have an exact word-to-word correspondence with the premise (*in the sand beside the ocean* $\rightarrow$ *on the beach*), but nevertheless DENEMBED$_+$ predicts a high probability. In example 7, we might guess that the patient is a man with 50% probability, so a predicted conditional probability around 0.5 seems reasonable. Example 14 shows that DENEMBED$_+$ can correctly predict very low conditional probability for sentences that have no topic in common, while example 13 shows a low predicted probability for a CONTRADICTION example that still has high word overlap.

There are also some examples in SNLI that illustrate why it may be difficult to apply strict thresholds to conditional probability in order to turn $\hat{P}_{[\![\,]\!]}(h|p)$ into an entailment class label. These examples are more ambiguous. In example 6, it is not certain that *people are playing music*, but it is a pretty reasonable assumption from the premise. Therefore it is not

surprising that the model assigns a higher conditional probability than for most NEUTRAL sentence pairs. In contrast, example 8 is also NEUTRAL, but yields a much lower predicted probability score. This is also a reasonable outcome, as *scoring a touchdown* is a relatively infrequent event in the space of an entire *football game*. It would be less reasonable to say that the probability of *scoring a touchdown* is fifty percent for any given moment of a football game.

DENEMBED$_+$ predictions are less accurate when the sentence structure differs substantially between premise and hypothesis, or when there are many unknown words, as in example 5 (*John Deere* is not in our vocabulary). Our model cannot reason about numbers and quantities, as example 10 shows (*two women* does not imply *three women*). It also fails to recognize in example 11 that a man *wearing a black shirt* is probably not *wearing a blue shirt* as well.

5.5.2 Negation

One shortcoming of our embedding model is that it does not allow us to represent the negation of x as a vector. We also cannot represent two phrases that have completely disjoint denotations: in Figure 5.1, the $P(x)$ and $P(y)$ regions will always intersect and therefore the $P(x, y)$ region will always have an area greater than 0. In fact, in our embedding space, the joint probability represented by the vector $\mathbf{z}$ will always be greater than or equal to the product of the probabilities represented by the vectors $\mathbf{x}$ and $\mathbf{y}$. For any pair $\mathbf{x} = (x_1, ..., x_N)$ and $\mathbf{y} = (y_1, ..., y_N)$, $\hat{P}_{\mathbb{\Pi}}(x, y) \geq \hat{P}_{\mathbb{\Pi}}(x)\hat{P}_{\mathbb{\Pi}}(y)$:

$$\hat{P}_{\mathbb{\Pi}}(x, y) = \exp\big(-\sum_i \max(x_i, y_i)\big) \tag{5.6}$$

$$\geq \exp\big(-\sum_i x_i - \sum_i y_i\big) \tag{5.7}$$

$$= \hat{P}_{\mathbb{\Pi}}(x)\hat{P}_{\mathbb{\Pi}}(y) \tag{5.8}$$

(Equality holds when $\mathbf{x}$ and $\mathbf{y}$ are orthogonal, and thus $\sum_i x_i + \sum_i y_i = \sum_i \max(x_i, y_i)$). Therefore, the best we can do for disjoint phrases is learn an embedding that assumes the phrases are independent. In other words, we can map the disjoint phrases to two vectors whose computed joint probability is the product of the individual phrase probabilities.

Although DENEMBED cannot represent two events with completely disjoint denotations, we have showed that it is still able to express that some phrase pairs have very low denotational conditional probabilities. We note also that DENEMBED cannot express $P(x) = 0$

exactly, but can get arbitrarily close in order to represent the probability of a phrase that is extremely unlikely.

Vilnis et al. [81] observe that this shortcoming is inherent to the class of models that assign probability measures to a partial order embedding: these models cannot capture negative correlation. Instead, they propose a new model that represents each concept as a high-dimensional product-of-intervals (also known as a *hyperrectangle* or *box*). In contrast to our model, which defines each phrase's probability as the forward cone of the vector, their model defines unary probabilities based on the box volume and joint probabilities from the overlap between boxes. Vilnis et al. show that their definition of box embeddings can model disjoint events, and empirically demonstrate this as well. The box lattice model produces improved results on WordNet hypernym prediction over Vendrov et al.'s order embedding model, and improved results on the DenPhrase probability predictions over our model. They do not evaluate on any further downstream tasks like textual entailment, so it remains to be seen whether the box lattice embeddings are informative to other semantic tasks.

5.6 CONCLUSION

In this chapter, we presented a framework for computing the denotational conditional probabilities between pairs of phrases from regions in a vector space that correspond to denotational set relationships. We demonstrated that we can successfully train a neural network model to predict these probabilities for new phrases. There are some limitations to the resulting phrase embeddings due to the limited vocabulary and short phrase lengths in the training data. However, we showed that by adding longer sentences with approximate probabilities to our training data, our embedding model can be adapted to standard textual entailment datasets, on which we demonstrate improved performance over a neural entailment baseline.

CHAPTER 6: MULTIPLE PREMISE ENTAILMENT

Standard textual entailment recognition is concerned with deciding whether one statement (the hypothesis) follows from another statement (the premise). So far in this thesis, we have only discussed tasks where both the premise and the hypothesis are single sentences. The sentence entailment task has encouraged the development of models that produce semantic representations for sentences independent of other natural language tasks like coreference resolution and named entity recognition. However, it has also resulted in entailment datasets that rely on local comparisons between two sentences, e.g. word matching and substitution of lexical synonyms or hypernyms.

In this chapter, we define Multiple Premise Entailment (MPE), a novel textual entailment task in which the premise text is an unordered set of independently written sentences that describe the same event. The premise sentences are all FLICKR30K captions from the same image (see examples in Table 6.1), so they may contain overlapping information but they are typically not paraphrases. Importantly, these premise sentences all share a common denotation which is the image they all describe. In light of this information, we want to learn more about how these sentences interact, how often they contain overlapping or new information, and so on.

There are many real-world situations in which we are presented with information from multiple perspectives. For example, we might read news articles from different sources in order to obtain a more complete picture of a reported story. Other examples are social media posts by different people about a single event, or multiple witness reports for a crime. In these cases, we want to use multiple independent reports to infer what really happened. The multiple FLICKR30K captions per image provide an ideal testbed for this task.

The hypothesis for this task is a simplified fifth FLICKR30K caption that may or may not come from the same image. Instead of soliciting humans to write new hypotheses, as Bowman et al. [4] did to build SNLI, we use simplified sentences from the denotation graph, and apply a word overlap filter and the graph structure to minimize the presence of trivial lexical relationships.

The task is to decide whether the hypothesis sentence a) can be used to describe the same scene (ENTAILMENT), b) cannot be used to describe the same scene (CONTRADICTION), or c) may or may not describe the same scene (NEUTRAL). The main challenge is to consider

Work in this chapter was first published in A. Lai, Y. Bisk, and J. Hockenmaier (2017), "Natural Language Inference from Multiple Premises," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing*, 100–109 [82]. It is reprinted here with the permission of the copyright holder.

all the premise sentences to infer what happened in the scene, in some cases aggregating information across multiple sentences into a coherent whole.

In this chapter, we describe how we collected a dataset for this task that minimizes trivial lexical inferences, emphasizes knowledge of everyday events, and presents a more challenging setting for textual entailment. We evaluate several strong neural baselines and analyze how the multiple premise task differs from standard textual entailment.

PREMISES: Two girls sitting down and looking at a book. A couple laughs together as they read a book on a train. Two travelers on a train or bus reading a book together. A woman wearing glasses and a brown beanie next to a girl with long brown hair holding a book. HYPOTHESIS: Women smiling. ⇒ENTAILMENT
PREMISES: Three men are working construction on top of a building. Three male construction workers on a roof working in the sun. One man is shirtless while the other two men work on construction. Two construction workers working on infrastructure, while one worker takes a break. HYPOTHESIS: A man smoking a cigarette. ⇒NEUTRAL
PREMISES: A group of individuals performed in front of a seated crowd. Woman standing in front of group with black folders in hand. A group of women with black binders stand in front of a group of people. A group of people are standing at the front of the room, preparing to sing. HYPOTHESIS: A group having a meeting. ⇒CONTRADICTION

Table 6.1: The Multiple Premise Entailment Task

6.1 CONSTRUCTING THE MPE DATASET

The MPE dataset[1] contains 10,000 items, each consisting of four premise sentences (captions from the same FLICKR30K image), one hypothesis sentence (a simplified FLICKR30K caption), and a label (ENTAILMENT, NEUTRAL, or CONTRADICTION) that indicates the relationship between the four premises and the hypothesis. The label is based on a consensus of five crowdsourced judgments. To analyze the difference between multiple premise and single

[1]Available to download at `https://github.com/aylai/MultiPremiseEntailment`.

premise entailment (Section 6.2.2), we also collected entailment labels for each individual premise-hypothesis pair in the development data.

6.1.1 Generating the Items

Hypothesis Simplification

The premise of each MPE item consists of four FLICKR30K captions from the same image, while the hypothesis sentence is a simplified caption. Since complete captions are very specific and are likely to introduce new details that are not entailed by the premises, we simplify the hypothesis caption according to the denotation graph generation process to produce a sentence that is more likely to be entailed by the premises. In 50% of our data, the hypothesis sentence is a simplified variant of the fifth caption describing the same image as the premise captions (this hypothesis is more likely to be entailed by the premises, but this is not guaranteed, which is why we have a human annotation step). In the remaining 50% of our data, the hypothesis sentence is a simplified variant of one of the captions for a random FLICKR30K image.

To simplify a hypothesis caption, we consider all sentence nodes in the denotation graph that are ancestors (more generic versions) of this caption, but exclude nodes that are also ancestors of any of the premises. This ensures that the simplified hypothesis sentence cannot be trivially obtained from any premise sentence via the same automatic simplification procedure. Therefore, we avoid some obvious semantic relationships between premises and hypothesis, such as hypernym replacement, dropping modifiers or prepositional phrases, etc.

Limiting Lexical Overlap

Given the set of simplified, restricted hypotheses, we further restrict the pool of potential items to contain only pairings where the hypothesis has a word overlap ≤ 0.5 with the premise set. We compute word overlap as the fraction of hypothesis tokens that appear in at least one premise (after stopword removal). This eliminates trivial ENTAILMENT cases where the hypothesis is simply a subset of the premise text. Table 6.2 shows that, as a result, the mean word overlap for the MPE training data is much lower than SNLI.

	SNLI		MPE	
Data	full	lemma	full	lemma
All	0.44 ± 0.29	0.48 ± 0.29	0.28 ± 0.22	0.33 ± 0.20
ENTAILMENT	0.59 ± 0.31	0.64 ± 0.30	0.34 ± 0.21	0.38 ± 0.19
NEUTRAL	0.41 ± 0.24	0.45 ± 0.24	0.28 ± 0.21	0.33 ± 0.19
CONTRADICTION	0.33 ± 0.25	0.36 ± 0.25	0.23 ± 0.22	0.30 ± 0.21

Table 6.2: Mean word overlap for full training data and each label, original and lemmatized sentences. MPE has much lower word overlap than SNLI.

Data Selection

From this constrained pool of premises-hypothesis pairings, we randomly sampled 8000 training items from the FLICKR30K training split. For test and development data, we sample 1000 items from FLICKR30K test and 1000 from dev. The hypotheses in the training data must be associated with at least two captions in the FLICKR30K train split, while the hypotheses in dev/test must be associated with at least two captions in the union of the train and dev/test splits, and with at least one caption in dev/test alone. Since the test and dev splits of FLICKR30K are smaller than the training split, this threshold is designed to select hypotheses that are rare enough to be interesting and frequent enough to be reasonable sentences.

We also limited repetition of hypothesis sentences to at most 16 times in the training set and at most two times each in the development and test splits to encourage diversity of hypotheses in the data.

6.1.2 Assigning Entailment Labels

The MPE task was inspired by the Approximate Textual Entailment (ATE) task [9], which also used a premise set of four FLICKR30K captions. ATE items were labeled automatically under the assumption that items were positive (approximately entailing) if the hypothesis came from the same image as the four premises, and negative otherwise. However, we ultimately found that this assumption was true for only half of the positive items. To address this discrepancy, we collected human judgments to label these items as ENTAILMENT, CONTRADICTION, or NEUTRAL.

Instructions:

We will show you four caption sentences that describe the same scene, and one proposed sentence. Your task is to decide whether or not the scene described by the four captions can also be described by the proposed sentence.

The four captions were written by four different people. All four people were shown the same image, and then wrote a sentence describing the scene in this image. Therefore, there may be slight disagreements among the captions. The images are photographs from Flickr that show everyday scenes, activities, and events. You will not be given the image that the caption writers saw.

Process:

Read the four caption sentences and then read the proposed sentence. Then choose a response to this question: *Can the scene described by the four captions also be described by the proposed sentence?*

YES: *The scene described by the captions can definitely (or very probably) be described by the proposed sentence.* The proposed sentence may leave out details that are mentioned in the captions. If the proposed sentence describes something that is not mentioned in the captions, it is probably safe to assume the extra information is true, given what you know from the captions. If there are disagreements among the captions about the details of the scene, the proposed sentence is consistent with at least one caption.

UNKNOWN: *There is not enough information to decide whether or not the scene described by the captions can be described by the proposed sentence.* There may be scenes that can be described by the proposed sentence and the captions, but you don't know whether this is the case here.

NO: *The scene described by the captions can probably not be described by the proposed sentence.* The proposed sentence and the captions either contradict each other or describe what appear to be two completely separate events.

Table 6.3: The annotation instructions we provided to CrowdFlower and Mechanical Turk annotators.

Crowdsourcing Procedure

For each item, we solicited five responses from workers on two crowdsourcing platforms, CrowdFlower and Mechanical Turk, as to whether the relationship between the set of four premises and the hypothesis was ENTAILMENT, CONTRADICTION, or NEUTRAL. Table 6.3 contains the instructions we provided to the workers.

The two crowdsourcing platforms differ somewhat in their interfaces and options to filter workers, so our annotation procedures differ slightly between CrowdFlower and Amazon. On CrowdFlower, we limited the annotator pool to the highest quality group of workers (as defined by CrowdFlower) and to countries with a high percentage of native English speakers. We also set a minimum time of 60 seconds per five questions. Annotators had to complete five questions to be paid for a job, and each annotator was allowed to complete a maximum of 800 questions. On Mechanical Turk, we only allowed workers located in the United States who had at least successful 1000 HITs with an overall approval rate of at least 95%. Annotators submitted one question at a time and there was no maximum on the number of questions that a single annotator could answer.

Entailment Labels

We assume three labels (ENTAILMENT, NEUTRAL, CONTRADICTION). For ENTAILMENT, we asked annotators to judge whether the hypothesis could *very probably* describe the same scene as the premises, rather than specifying that the hypothesis must *definitely* be true, as Bowman et al. [4] did for SNLI. We also told annotators that the hypothesis could mention something not described in the premises as long as most people would assume it was true. Our instructions align with the standard definition of textual entailment: "T entails H if humans reading T would typically infer that H is most likely true" [5]. We are interested in more than what is logically required for a hypothesis to be true: our goal was to solicit examples of entailment that depend on people's assumptions about the world that are rarely stated directly.

SNLI does contain some ENTAILMENT examples where the hypothesis is not strictly required to be true from the premise. For example:

PREMISE: A woman in a tan top and jeans is sitting on a bench wearing headphones.
HYPOTHESIS: A woman is listening to music.

⇒ENTAILMENT

or

PREMISE: A dog running in the sand.
HYPOTHESIS: A dog is running outside at the beach.

⇒ENTAILMENT

Both of the above examples are labeled ENTAILMENT even though one can imagine scenarios where the premise is true while the hypothesis is not. These examples might not be as unambiguous as the strict entailment example below, but they are also more interesting:

PREMISE: Two men on bicycles competing in a race.
HYPOTHESIS: People are riding bikes.

⇒ENTAILMENT

Our goal was to include entailment examples that are more interesting and natural, and reflect human assumptions about language and correlation in the world.

Examples

We provided a small set of labeled example items to the annotators along with an explanation for the label. This text is reproduced here:

PREMISES:
A man in jean shorts and a dark shirt is sitting with a red backpack beside him in a wooded area possibly chanting.
A man with a headband sitting on a green bench.
A man takes a break from his hiking trip.
An older man sitting in the sunshine.

HYPOTHESIS: A hiker sitting. ⇒ENTAILMENT

In the above example, the captions describe a man on a hike, so *hiker* is an appropriate description. Three of the four captions say the man is sitting and the fourth caption doesn't contradict this information, so the proposed sentence describes the scene.

PREMISES:
A thin man talks in front of a seated crowd and explains something with his hands.
A man in a dark shirt and orange lanyard speaking to an audience.
A man is giving a presentation in front of a crowd.
A man is giving a presentation.

HYPOTHESIS: A man standing. ⇒ENTAILMENT

In this second example, we assume that a man who is giving a presentation is standing unless otherwise mentioned, so the proposed sentence is a true description.

PREMISES:
Three girls sitting at a dinner eating and looking around.
Three children are sitting down at a table eating.
Four girls eat breakfast at a convention.
Three children sit down to a meal.

HYPOTHESIS: Three girls sitting at a table. ⇒ENTAILMENT

Although the captions disagree about whether there are three or four girls in the scene, the proposed sentence is consistent with one of these possible scenarios, so it describes the scene. A different proposed sentence *"Four girls sitting at a table"* would also describe the scene.

PREMISES:
People in a school cafeteria with a boy in the foreground wearing yellow and brown stripes.
A crowd of people gather for a meal indoors.
Children serving at a community dinner.
A room full of adults and children.

HYPOTHESIS: A child does dishes. ⇒NEUTRAL

It is quite possible that there is a child doing dishes at a dinner served by children, but none of the captions mention anyone doing dishes. There is not enough information to decide that the proposed sentence either does or does not describe the scene.

PREMISES:
There are two girls dressed in exercise clothing obviously doing an exercise on a beach.
Two young women, both with long brown hair, practicing yoga on a rocky beach.
Two females are working out on a beach.
Two women, on a beach doing yoga.

HYPOTHESIS: Women stand. ⇒NEUTRAL

The captions describe two women in the scene, but we don't know if they are standing because yoga involves both sitting and standing positions. There is not enough information to decide that the proposed sentence either does or does not describe the scene.

PREMISES:
Two men are sitting in white chairs next to a blue door.
Two men in white plastic chairs sitting in a doorway.
Two elderly men converse in an run-down building.
Two men are sitting on white chairs.

HYPOTHESIS: A man does crunches. ⇒CONTRADICTION

Neither of the men described in the captions can be doing crunches because they are sitting in chairs. It is unlikely that there is a third man in the same scene who is doing crunches, so the proposed sentence probably does not describe the scene.

PREMISES:
Eight people are shown in the picture in snow gear seeming to be skiing.
Seven people are taking a break from skiing to chat in a snowy clearing.
Several skiers standing around talking in a snowy scene.
A group of adults getting ready to ski.

HYPOTHESIS: A group swimming. ⇒CONTRADICTION

It does not make sense for a group of people to be swimming in a skiing scene, so the proposed sentence is not compatible with the scene described by the captions.

Final Label Assignment

Of the 10,000 items for which we collected annotations, 90% had a majority label based on the five judgments, including 16% with a 3-2 split between ENTAILMENT and CONTRADICTION. The remaining 10% had a 2-2-1 split across the three classes. We manually adjudicated the latter two cases.

The dataset we released contains both our final labels and the crowdsourced judgments for all items. 82% of the final labels in the dataset agree with the majority voting label over the submitted judgments (the remaining 18% differ due to our manual correction). The disagreements are due to various factors. In some cases, the item is simply more difficult and therefore disagreement among annotators is to be expected for these edge cases. In other cases, some annotators provided questionable judgments. We did attempt to weed out unskilled or malicious annotators, and collected five judgments per item to alleviate the affects of bad annotators, but some of these disagreements are inevitable in any crowdsourced data.

6.1.3 Image IDs

Each premise sentence in MPE has a corresponding FLICKR30K image ID (included in the data release). Since we are interested primarily in the information present in linguistic descriptions of a scene, the labels reflect the text-based entailment relationship between the premise text and the hypothesis. However, future work could apply multi-modal representations to this task, with the caveat that including the image would likely resolve many NEUTRAL items to either ENTAILMENT or CONTRADICTION. We explore a version of this task in Chapter 7.

6.2 DATASET CHARACTERISTICS

6.2.1 Statistics

The dataset contains 8000 training items, 1000 development items, and 1000 test items. Table 6.4 shows overall type and token counts and sentence lengths as well as the label distribution.

The mean annotator agreement, i.e. the fraction of annotators who agreed with the final label, is 0.70 for the full dataset (0.82 for ENTAILMENT, 0.42 for NEUTRAL, and 0.78 for CONTRADICTION). That is, on average, four of the five crowdsourced judgments agree

	SNLI	MPE
Number of Lexical types	36,616	9,254
Number of Lexical tokens	12 million	468,524
Mean premise length	14.0 ± 6.0	53.2 ± 12.8
Mean hypothesis length	8.3 ± 3.2	5.3 ± 1.8
Label distribution		
ENTAILMENT	33.3%	32.3%
NEUTRAL	33.3%	26.3%
CONTRADICTION	33.3%	41.6%

Table 6.4: Type and token counts, sentence lengths, and label distributions for the training data.

with the final label for ENTAILMENT and CONTRADICTION. For NEUTRAL, only two of the five original annotators on average selected NEUTRAL, and the other three selected either CONTRADICTION or ENTAILMENT.

6.2.2 MPE vs. Standard Entailment

Multiple premise entailment differs from standard single premise entailment (SPE) in that each premise consists of four independently written sentences about the same scene. To understand how MPE differs from SPE, we collected pairwise single-premise entailment labels for each individual premise-hypothesis pair in the development data. Each pair label is based on three human judgments, again collected using Mechanical Turk.

In Table 6.5, we compare the full item MPE entailment labels to the four pair SPE labels. The number of SPE labels that agree with the MPE label yields the five categories in Table 6.5, ranging from the most difficult case where none of the SPE labels agree with the MPE label (21.8% of the data) to the simplest case where all four SPE labels agree with the MPE label (9.8% of the data).

We observe that a simple majority voting scheme over the gold standard SPE labels is not sufficient to do well on MPE, since it assigns the correct MPE label to only 34.6% of the development items (i.e. those cases where three or four SPE pairs agree with the MPE label). We also consider a slightly more sophisticated voting scheme that applies the following heuristic (here, $|E|$ and $|C|$ are the number of SPE ENTAILMENT and CONTRADICTION labels of each class):

# pairs agree	% of data	Pair Label	Example **Hypothesis** and Four Premises
0	21.8	N	A football player in a red uniform is standing in front of other football players in a stadium.
		N	A football player facing off against two others.
		N	A football player wearing a red shirt.
		N	Defensive player waiting for the snap.
			The team waiting.
			⇒ENTAILMENT
1	26.9	N	A person is half submerged in water in their yellow kayak.
		C	A woman has positioned her kayak nose down in the water.
		N	A person in a canoe is rafting in wild waters.
		N	A kayaker plunges into the river.
			A man in a boat paddling through waters.
			⇒CONTRADICTION
2	16.7	E	A batter playing cricket missed the ball and the person behind him is catching it.
		E	A cricket player misses the pitch.
		N	The three men are playing cricket.
		N	A man struck out playing cricket.
			A man swings a bat.
			⇒ ENTAILMENT
3	24.8	N	A young gymnast, jumps high in the air, while performing on a balance beam.
		N	A gymnast performing on the balance beam in front of an audience.
		E	The young gymnast's supple body soars above the balance beam.
		N	A gymnast is performing on the balance beam.
			A woman doing gymnastics.
			⇒ NEUTRAL
4	9.8	C	A man with a cowboy hat is riding a horse that is jumping.
		C	A cowboy riding on his horse that is jumping in the air.
		C	A cowboy balances on his horse in a rodeo.
		C	Man wearing a cowboy hat riding a horse.
			Men pulled by animals.
			⇒ CONTRADICTION

Table 6.5: MPE examples categorized by the number of pairs labels (0–4) that agree with the full label.

If $|E| > |C|$, predict ENTAILMENT.
Else if $|C| > |E|$, predict CONTRADICTION.
Otherwise, predict NEUTRAL.

This baseline achieves 41.7% accuracy, indicating that MPE cannot be trivially reduced to SPE. That is, even if a model could predict the correct SPE label for each individual premise with 100% accuracy (an unrealistic assumption), it would require more than simple voting heuristics to obtain the correct MPE label from the pairwise labels.

Table 6.5 illustrates that the majority of MPE items require aggregation of information about entities and events from multiple premises. In the first example, the first premise is consistent with a scene that involves a team of football players but does not indicate what the team is doing, while the last premise indicates that a football player (and therefore the rest of the team) is waiting, but does not actually mention the team.

6.2.3 Semantic Phenomena

We used a random sample of 100 development items to examine the types of semantic phenomena that are useful for inference in MPE. We categorized each item by the type of knowledge or reasoning necessary to predict the correct label for the hypothesis given the premises. An item belongs to a category if at least one premise exhibits the specified phenomenon in relation to the hypothesis, and an item may belong to multiple categories. Table 6.6 shows the frequency of each category, an illustrative example containing the relevant premise, and the distribution over entailment labels. We used all four premises for our analysis, but we show only a single relevant premise along with the hypothesis for simplicity.

Word equivalence Items in this category contain a pair of equivalent words (synonyms or paraphrases). The word in the hypothesis can be exchanged for the word in the premise without significantly changing the meaning of the hypothesis.

Word hypernymy These items involve lexical hypernyms: either one word is the hypernym of the other, or they share a mutual hypernym. For example, a *man* is also a *person* (ENTAILMENT), but a *person* may or may not be a *man* (NEUTRAL), and somebody who is a *man* is not a *child* (CONTRADICTION).

Phrase equivalence These items involve equivalent phrases, i.e. synonyms or paraphrases. The phrase in the hypothesis can be replaced by the phrase in the premise without

	#	E	N	C	Example Premise and **Hypothesis** Pair
Total	100	31	29	40	
Word equivalence	16	12	4	0	A person *climbing* a rock face. **A rock climber *scales* a cliff.** ⇒ENTAILMENT
Word hypernymy	19	6	6	7	*Girl* in a blue sweater painting while looking at a bird in a book. **A *child* painting a picture.** ⇒ENTAILMENT
Phrase equivalence	7	6	1	0	*A couple in their wedding attire* stand behind a table with a wedding cake and flowers. ***Newlyweds* standing.** ⇒ ENTAILMENT
Phrase hypernymy	8	6	2	0	A group of young boys wearing track jackets *stretch their legs* on a gym floor as they sit in a circle. **A group *doing exercises.*** ⇒ ENTAILMENT
Mutual exclusion	25	0	0	25	A woman in a red vest *working at a computer.* **Lady *doing yoga.*** ⇒ CONTRADICTION
Compatibility	18	0	18	0	*Onlookers watch.* **A girl at bat in a *softball game.*** ⇒ NEUTRAL
World knowledge	35	14	9	12	A young woman gives directions to an older woman outside a subway station. **Women standing.** ⇒ ENTAILMENT

Table 6.6: Analysis of 100 random dev items. For each phenomenon, we show the distribution over labels and an example. We use italics to indicate the relevant comparisons. The indicated span of text is part of the necessary information to predict the correct label, but may not be sufficient on its own.

significantly changing the meaning of the hypothesis. The phrases cannot be decomposed into multiple lexical comparisons.

Phrase hypernymy Items in this category involve a specific phrase and a general phrase: the more general phrase *doing exercises* can refer to multiple types of exercises in addition to *stretching their legs*.

Mutual exclusion Distinguishing between CONTRADICTION and NEUTRAL involves identifying actions that are mutually exclusive, i.e. cannot be performed simultaneously by the same agent (*"Two doctors perform surgery"* vs. *"Two surgeons are having lunch"*). We previously demonstrated in Section 4.4.1 that this kind of information encoded in an antonym

dictionary was particularly useful for identifying CONTRADICTION in SICK.

Compatibility The opposite of mutual exclusion is compatibility: when two actions can be performed simultaneously by the same agent (e.g. *"A boy flying a red and white kite"* vs. *"A boy is smiling"*).

World knowledge These items require extra-linguistic knowledge about the relative frequency and co-occurrence of events in the world (not including the mutual exclusion or compatibility phenomena). For example, a human reader can infer that *children in a potato sack race* are *having fun* (while *a marathon runner competing in a race* might not be described as *having fun*).

When we look at the distribution of these phenomena as shown in Table 6.6, we see that many items (19–35%) involve lexical relationships, either lexical synonyms or hypernyms. These types of relationships are also common in SNLI, and it is not surprising that they appear frequently in MPE as well, due to the construction of the image caption data and the denotation graph: the multiple captions from FLICKR30K encourage the presence of synonyms, and the denotation graph phrases are generated using WordNet hypernym information. However, in addition to these lexical relationships, we see a nontrivial number of items (8–15%) that involve phrase comparisons that cannot be easily decomposed into words. These phrasal synonyms or hypernyms are interesting as they differ from most constructions in SNLI.

Finally, we have the mutual exclusion and compatibility categories, which are a specific type of world knowledge. These three categories all involve some external information about the distribution of events in the world: certain actions cannot be done simultaneously by the same person, while this is not true of other action pairs. We consider these items to be the most difficult in our dataset, and together they comprise 35-78% of the data, a significant portion of the items in MPE. Modeling these phenomena should present an interesting challenge for future work.

6.2.4 Combining Information Across Premises

In addition to the semantic phenomena we have just discussed, MPE presents the challenge of how to combine information across multiple premises. We examined examples from the development data to categorize the different types of information aggregation present in our dataset.

Coreference resolution These items require cross-caption coreference resolution of entity mentions from multiple premises and the hypothesis. In this example, *two men* and *two senior citizens* refer to the same entities, i.e. the *two older men* in the hypothesis. Given that information and other premise descriptions, the reader can infer that the two older men on the street are likely to be standing.

PREMISES:
Two men in tan coats exchange looks on the city sidewalk.
Two senior citizens talking on a public street.
Two men in brown coats on the street.
Two men in beige coats, talking.

HYPOTHESIS: Two older men stand. ⇒ENTAILMENT

Event resolution This case requires resolving various event descriptions from multiple premises and the hypothesis. In the following example, a human reader recognizes that the man is sitting on scaffolding so that he can repair the building, and therefore he is doing construction work.

PREMISES:
A man is sitting on a scaffolding in front of a white building.
A man is sitting on a platform next to a building ledge.
A man looks down from his balcony from a stone building.
Repairing the front of an old building.

HYPOTHESIS: A man doing construction work. ⇒ENTAILMENT

Visual ambiguity resolution This case involves reconciling apparently contradictory information across premises. These discrepancies are largely due to the fact that the premise captions were written about an image. Sometimes the image contained visually ambiguous entities or events that are then described differently by different caption writers. In this example, in order to resolve the discrepancy, the reader must recognize from context that *woman* and *young child* (and also *person*) refer to the same entity.

PREMISES:
A person in a green jacket and pants appears to be digging in a wooded field with several cars in the background.
A young child in a green jacket rakes leaves.
A young child rakes leaves in a wooded area.
A woman cleaning up a park.

HYPOTHESIS: A woman standing in the forest. ⇒ENTAILMENT

Scene resolution These examples require the reader to build a mental representation of the scene from the premises in order to assess the probability that the hypothesis is true. In the first example, specific descriptions – *a jumping horse*, *a cowboy balancing*, *a rodeo* – can be combined to conclude that there is a high probability that the specific event described by the hypothesis is true.

PREMISES:
A man with a cowboy hat is riding a horse that is jumping.
A cowboy riding on his horse that is jumping in the air.
A cowboy balances on his horse in a rodeo.
Man wearing a cowboy hat riding a horse.

HYPOTHESIS: An animal bucking a man. ⇒ENTAILMENT

In the next example, the hypothesis does not contradict any individual premise sentence. However, a reader who understands the generic scene described knows that the very specific hypothesis description is unlikely to go unmentioned if it were a true description of the premise scene. Shirtlessness would be a salient detail in this scene, so the fact that none of the premises mention it means that the hypothesis is likely to be false.

PREMISES:
A young couple sits in a park eating ice cream as children play and other people enjoy themselves around them.
Couple in park eating ice cream cones with three other adults and two children in background.
A couple enjoying ice cream outside on a nice day.
A couple eats ice cream in the park.

HYPOTHESIS: A shirtless man sitting. ⇒CONTRADICTION

In the final example, the premises present a somewhat generic description of the scene. While some premises lean towards ENTAILMENT (*a woman and a man in discussion* could

be *having a work meeting*) and others lean towards CONTRADICTION (*two people conversing outdoors at a restaurant* are probably not *working*), none of them contain overwhelming evidence that the scene entails or contradicts the hypothesis. Therefore, the hypothesis is NEUTRAL given the premises.

PREMISES:
A blond woman wearing a gray jacket converses with an older man in a green shirt and glasses while sitting on a restaurant patio.
A blond pony-tailed woman and a gray-haired man converse while seated at a restaurant's outdoor area.
A woman with blond hair is sitting at a table and talking to a man with glasses.
A woman discusses something with an older man at a table outside a restaurant.

HYPOTHESIS: A woman doing work. ⇒NEUTRAL

6.2.5 Hypothesis-Only Bias

One of our motivations for creating this dataset was the idea that SNLI contains certain biases due to how Bowman et al. [4] elicited hypotheses from annotators. We aimed to avoid some of these issues by using randomly selected hypotheses that were derived from image captions independently of the entailment label.

Recent work has verified this hypothesis: both Gururangan et al. [11] and Poliak et al. [12] showed that a hypothesis-only model, which encodes only the hypothesis of each sentence pair, achieves almost 70% accuracy on SNLI. This is compared to a majority-class baseline of 33%. In other words, a significant fraction of the test set can be correctly classified without knowledge of the premise sentence. Both papers demonstrate that some of this bias is encoded at the word level: certain words and types of words are more likely to occur with a particular entailment label. Gururangan et al. hypothesize that these artifacts may result from annotators using heuristics in order to quickly write hypotheses for a specific entailment label.

We took a *human judgment* approach to our dataset rather than *human elicitation*. We constructed the items in our dataset by selecting hypothesis sentences from the denotation graph and randomly pairing them with premises. Poliak et al. show that we were successful in avoiding this bias in the hypothesis, as their hypothesis-only model applied to MPE shows no improvement in accuracy over the majority class baseline. However, it is not clear what part of our data selection and annotation process is responsible for this result; Poliak et al. show that other entailment datasets *without* human-elicited hypotheses still show some

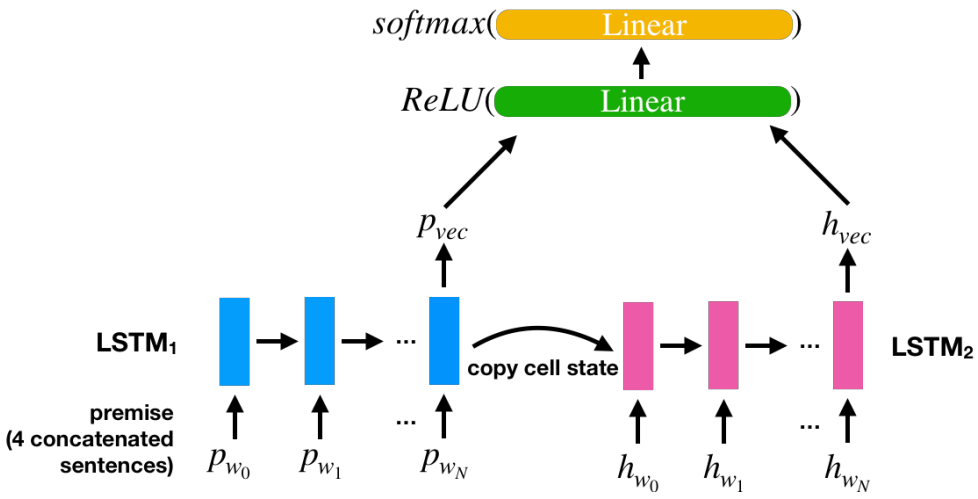


Figure 6.1: LSTMCond: an entailment classifier that uses two LSTM RNNs to process the hypothesis conditioned on the premise sentence(s). To apply this model to MPE, we concatenate the four premises into a single sequence that is processed by the first LSTM.

hypothesis-only bias.

Hypothesis-only bias may exist for many reasons, e.g. sentence lengths, the previously mentioned issue of lexical choice, or unexplored phenomena like syntax or multi-word expressions. Furthermore, hypothesis-only bias is not the only possible type of bias in a textual entailment dataset. MPE may not contain label biases due to the hypothesis, but that does not mean it is free from other biases. It is important to continue these investigations into potential dataset bias to better understand the models we train.

6.3 NEURAL ENTAILMENT MODELS

We apply several neural models from the entailment literature to predict entailment on MPE. We also present a model designed to handle multiple premises, as this is unique to our dataset.

6.3.1 A Conditional LSTM Classifier

In our experiments, we found that an entailment model that uses two LSTMs where one conditioned is on the other [46] outperformed an entailment classifier with a single LSTM (e.g. Bowman et al. [4], EntailLSTM from Chapter 5). We refer to this conditional LSTM classifier as LSTMCond (Figure 6.1). The model consists of two LSTMs, one to process the premise, and the second to process the hypothesis conditioned on the premise. After

the first LSTM reads the premise, its final cell state is used to initialize the cell state of the second LSTM, which reads the hypothesis. The resulting premise vector and hypothesis vector are concatenated and passed through a hidden layer and a softmax prediction layer. When handling four MPE premise sentences, we concatenate them into a single sequence (ordered by their caption IDs) that we pass to the first LSTM.

6.3.2 Word-to-Word Attention

Neural attention models have been very successful on SNLI, so we also consider a word-to-word attention model [46].[2] This model, which we refer to as ATTENTION, learns a soft alignment of words in the premise and hypothesis. One LSTM reads the premise and produces output vectors $\mathbf{Y} = [\mathbf{p}_1 ... \mathbf{p}_L]$ for the L words in the premise. The second LSTM processes the hypothesis one word at a time. For each word x_t in the hypothesis, the attention mechanism produces a vector of weights α_t by attending to all the premise output vectors $\mathbf{p}_1 ... \mathbf{p}_L$ and using the $\mathbf{h}_t$, the hypothesis LSTM output for this word, and the weighted premise representation from the previous timestep, $\mathbf{r}_{t-1}$.

$$\mathbf{M}_t = \tanh\left(\mathbf{W}^y \mathbf{Y} + \left(\mathbf{W}^h \mathbf{h}_t + \mathbf{W}^r \mathbf{r}_{t-1}\right) \otimes \mathbf{e}_L\right) \tag{6.1}$$

$$\alpha_t = \text{softmax}\left(\mathbf{w}^T \mathbf{M}_t\right) \tag{6.2}$$

$$\mathbf{r}_t = \mathbf{Y}\alpha_t^T + \tanh\left(\mathbf{W}^t \mathbf{r}_{t-1}\right) \tag{6.3}$$

The final sentence pair representation is a nonlinear combination of the final attention-weighted representation of the premise $\mathbf{r}_L$ and the final output vector from the hypothesis LSTM, $\mathbf{h}_N$. This final sentence pair representation is passed through a softmax layer to compute the cross-entropy loss.

$$\mathbf{h}^* = \tanh\left(\mathbf{W}^p \mathbf{r}_L + \mathbf{W}^x \mathbf{h}_N\right) \tag{6.4}$$

In the above equations, $\mathbf{e}_L$ is a vector of ones. $\mathbf{W}^y$, $\mathbf{W}^h$, $\mathbf{W}^r$, $\mathbf{W}^t$, $\mathbf{W}^p$, and $\mathbf{W}^x$ are all learned parameter matrices.

When training on MPE, we concatenate the premise sentences into a single sequence as the input to the premise LSTM.

[2] We use a reimplementation of Rocktäschel et al.'s model (`https://github.com/junfenglx/reasoning_attention`).

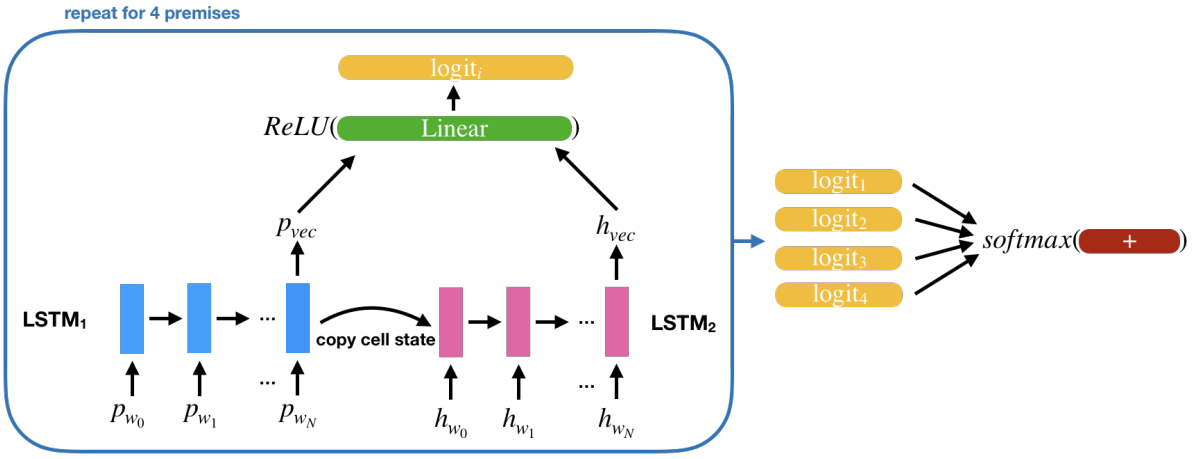


Figure 6.2: SumOfExperts: an entailment classifier that uses a pair of LSTM RNNs to process the hypothesis sentence four times conditioned on each premise sentence, and then aggregates the predictions.

6.3.3 Premise-wise Sum of Experts (SE)

Both LSTMCond and Attention assume that the premise is a single sentence, so we have to concatenate the four premises in order to apply the models to MPE. To capture what distinguishes MPE from standard entailment, we also consider a premise-wise sum of experts model (SumOfExperts, Figure 6.2) that makes a separate entailment prediction for each premise paired with the hypothesis. This model can adjust how it handles each premise based on the predictions of the other premises.

We use the same conditional LSTM setup as in LSTMCond and apply it repeatedly to read each premise and the hypothesis, producing four premise vectors $p_1 ... p_4$ and four hypothesis vectors $h_1 ... h_4$ (conditioned on each premise). Each premise vector p_i is concatenated with its hypothesis vector h_i and passed through a feedforward layer to produce logit prediction l_i. We sum $l_1 ... l_4$ to obtain the final prediction, which we use to compute the cross-entropy loss.

When training SumOfExperts on SNLI, we use the conditional LSTM only once to read the premise and hypothesis and produce p_1 and h_1. We pass the concatenation of p_1 and h_1 through the feedforward layer to produce l_1, which we use to compute the cross-entropy loss.

6.4 EXPERIMENTAL RESULTS

6.4.1 Training Details

We use the Adam optimizer with a learning rate of 0.001 for all models. We train each model for 10 epochs based on convergence on the development data. For joint SNLI+MPE training, we pretrain the model for 10 epochs on SNLI, then train for 10 epochs on MPE. This was the best joint training approach we found, compared to other approaches that involved balancing the SNLI and MPE data sizes, or interleaving the SNLI and MPE training epochs.

For all experiments that involve training on SNLI (either as the sole training data or as a pretraining step), we apply the best reported parameters for the ATTENTION model to train all three of our models: dropout keep rate 0.8, LSTM dimensionality 100d, batch size 32. For the experiments that involve training only on MPE, we chose hyperparameters from a grid search. For LSTMCOND: dropout keep rate 0.8, LSTM dimensionality 75d, batch size 32. For SUMOFEXPERTS: dropout keep rate 0.8, LSTM dimensionality 100d, batch size 32. For ATTENTION: dropout keep rate 0.6, LSTM dimensionality 100d, batch size 32.

6.4.2 Word Embedding Representations

For LSTMCOND and SUMOFEXPERTS, we use 300d GloVe vectors (trained on 840B tokens) as the word embedding input. Bowman et al. [4] presented some of the first competitive neural network models for textual entailment, and established GloVe vectors as the default embedding representation for neural entailment models. Later works that also evaluate on SNLI have followed this precedent in using GloVe vectors [43, 45, 47, 38, 83]. However, the ATTENTION implementation uses word2vec vectors.

In Table 6.7, we compare the effects of GloVe vs. word2vec on ATTENTION's accuracy on the development data. For the training regimes involving SNLI, word2vec vectors outperform GloVe vectors, while GloVe vectors slightly outperform word2vec when only training on MPE. These preliminary results are not necessarily reflective of the performance of GloVe vs. word2vec on SNLI or MPE in general. Prior work has shown that the performance of different word vectors may be task-specific and is also subject to sometimes hidden design decisions like preprocessing steps and hyperparameter choices [84, 85, 86, 87]. In the experiments in this chapter, we use word2vec vectors for ATTENTION under the assumption that its parameters were tuned for that embedding representation, but we use GloVe vectors for LSTMCOND and SUMOFEXPERTS to maintain continuity with the entailment community.

Vector source	Training regime		
	SNLI	MPE	SNLI+MPE
word2vec	54.3	54.0	64.0
GloVe	48.1	55.6	60.3

Table 6.7: Comparing entailment accuracy of ATTENTION using GloVe vs. word2vec vectors, evaluated on MPE development data.

Training	Class	LSTMCOND	SUMOFEXPERTS	ATTENTION
SNLI only		52.6	**55.9**	55.0
	E	85.8	71.5	81.7
	N	8.4	21.6	16.4
	C	55.7	62.0	54.5
MPE only		53.5	**56.3**	53.9
	E	63.1	61.3	48.3
	N	39.2	30.2	30.6
	C	53.5	66.5	71.2
SNLI+MPE		60.4	60.0	**64.0**
	E	65.1	65.4	75.9
	N	40.9	42.7	32.8
	C	67.2	65.1	71.5

Table 6.8: Entailment test accuracy on MPE. SUMOFEXPERTS has the highest accuracy of all models that are trained only on SNLI or MPE. ATTENTION has the highest accuracy when pretrained on SNLI.

6.4.3 Classification Results

Table 6.8 contains the test accuracies of the three models under three training regimes: SNLI only, MPE only, and SNLI+MPE.

We train only on SNLI to see whether models can generalize from one entailment task to the other. Both LSTMCOND and SUMOFEXPERTS are able to transfer information learned on SNLI to MPE; their performance does not degrade much from the MPE-only training regime to the SNLI-only regime. In contrast, however, ATTENTION's does much better on MPE when it is trained only on SNLI, compared to training in-domain only on MPE. We hypothesize that this is because ATTENTION has many more parameters than either LSTMCOND or SUMOFEXPERTS, and MPE alone does not contain sufficient data to train reliably. SNLI, however, does contain enough data for ATTENTION to learn reasonable parameters, enough to perform adequately on MPE. Its performance in the SNLI-only setting

(55.0%) still does not measure up to SumOfExperts trained on MPE (56.3%), which is trained on much less data.

The model with the highest accuracy after training in-domain is SumOfExperts (56.3%). It also outperforms all models that were trained only on SNLI. SumOfExperts is the only model of the three that processes the four premises as individual sentences and not as a single concatenated sequence, which appears to be an important distinction. SumOfExperts's strong performance supports our hypothesis that different premises may contain different information that needs to be combined carefully with other premises in order to succeed on this task.

LSTMCond performs on par with SumOfExperts when training on SNLI+MPE, but our analysis (Section 6.4.5) shows that their errors are quite different. LSTMCond's comparatively poor performance is not surprising as it is forced to reduce a very long sequence of words to a single vector without the benefit of attention to highlight important words.

Overall, the best performing model under any training regime is Attention trained on SNLI+MPE. We hypothesize that pretraining Attention on SNLI is necessary to learn reasonable parameters before training on MPE, a smaller dataset where word-to-word inferences may be less obvious. Attention's relative success confirms our analysis in Table 6.6 that there are a substantial minority of items in this dataset that contain lexical relationships similar to those found in SNLI. However, even the most successful model achieves only 64% accuracy on our dataset, compared to the 83.5% that Attention reaches on SNLI, so there is still substantial room for improvement on this dataset.

There are benefits both from using word-to-word attention and from handling the premises as individual sentences as SumOfExperts does. These benefits are potentially complementary and could be explored in future work. We did implement a preliminary model that added attention to the SumOfExperts architecture, but it overfit on SNLI and could not match other models' accuracy, reaching only about 58% on the development data compared to 59-63% from the other models.

6.4.4 Pair Agreement Results

We analyze how each model's performance is affected by the number of premises whose SPE label agrees with the MPE label. Table 6.9 shows the accuracy of each model (trained on SNLI+MPE) on the development data grouped by SPE-MPE label agreement (using the categories defined in Table 6.5).

Attention and SumOfExperts exhibit different strengths: Attention has the highest accuracy on three of five categories, including the most difficult category where none of the

Accuracy on SPE-MPE agreement subsets					
# pairs agree	0	1	2	3	4
% of data	21.8	26.9	16.7	24.8	9.8
LSTMCond	57.3	57.6	60.5	67.1	63.3
SumOfExperts	59.6	**58.0**	**63.3**	62.9	66.3
Attention	**65.6**	57.6	62.9	**68.3**	**70.4**

Table 6.9: Accuracy for each model (trained on SNLI+MPE) on the development data split according to how many SPE labels match the MPE label (Table 6.5).

	Accuracy			
Phenomenon	LSTMCond	SumOfExperts	Attention	#
Word equivalence	50.0	56.2	**68.8**	16
Word hypernymy	**52.6**	47.4	**52.6**	19
Phrase equivalence	57.1	57.1	**85.7**	7
Phrase hypernymy	50.0	50.0	**62.5**	8
Mutual exclusion	68.0	**72.0**	60.0	25
Compatibility	50.0	**61.1**	50.0	18
World knowledge	57.1	**62.9**	45.7	35

Table 6.10: Accuracy across semantic phenomena on 100 annotated development items. While Attention was the best model overall, it does not have the highest accuracy in each category.

SPE labels match the MPE label, while SumOfExperts has the highest accuracy in the remaining two categories. Attention demonstrates large gains in the easiest categories, perhaps because there is less advantage to aggregating individual premise predictions (as SumOfExperts does) and using attention to reweight individual words is more useful. On the other hand, Attention also does well on the most difficult category, indicating that it may be able to partially aggregate premise information by increasing attention weights on phrases from multiple sentences.

These results again emphasize that the SumOfExperts and Attention models might be complementary and have the potential to produce even better results on MPE when combined.

6.4.5 Semantic Phenomenon Results

Table 6.10 shows the performance of the three SNLI+MPE-trained models over semantic phenomena, based on the 100 annotated dev items (see Section 6.2.3 and Table 6.6). The

smaller categories (e.g. phrase equivalence and phrase hypernymy) may contain too few examples to be informative, but other categories still provide some information about the models.

Although ATTENTION outperformed both LSTMCOND and SUMOFEXPERTS in overall accuracy, it does not have the highest accuracy in every semantic category. Both SUMOFEXPERTS and ATTENTION have access to the same information, but ATTENTION does better on items that contain hypernyms and synonyms for both words and short phrases. Meanwhile, SUMOFEXPERTS is best at mutual exclusion, compatibility, and world knowledge categories, e.g. knowing that a man who is *resting* is not *kayaking*, and a *bride* is not also a *cheerleader*. In cases that require analysis of mutually exclusive or compatible events, a model like SUMOFEXPERTS has an advantage since it can reinforce its weighted combination prediction by examining each premise separately.

6.5 CONCLUSION

In this chapter, we presented a novel textual entailment task that involves inference over longer premise texts and aggregation of information from multiple independent premise sentences. Each premise text in our corpus contains multiple sentences that share a common denotation – the single image that they all describe – and the ultimate goal is to model a unified scene representation that can be used to judge the entailment relation between premises and hypothesis.

We presented several strong neural entailment baselines for this new task, including one model that aggregates information from the predictions of separate premise sentences. In the next chapter, we will continue this exploration by applying a denotational embedding model to this task.

CHAPTER 7: DENOTATIONAL EMBEDDINGS FOR MULTIPLE PREMISE ENTAILMENT

In Chapter 6, we introduced the MPE dataset as a new challenge for textual entailment models. MPE highlights phenomena that are not as common in other entailment datasets, including compatibility: how likely it is that two events occur simultaneously in the same setting. We have already presented the results of several reasonably successful neural entailment models. Moving forward, we want to apply denotational similarity to this task, as it expresses exactly the information that MPE requires: the likelihood that two phrases are both valid descriptions of an event.

In this chapter, we introduce a new model for learning denotational phrase embeddings and applying them to textual entailment. Our new denotational phrase embedding model is not interpretable, but it produces representations that are more flexible and informative for downstream tasks. We demonstrate that denotational information is useful to textual entailment in MPE and that denotational embeddings outperform standard neural sequence models.

We present several analyses of the successes and failures of our new model. We compare its errors to an SNLI-pretrained model to see the effects of different pre-training data. We also examine the characteristics of the denotational embedding model: what aspects of the denotation graph contribute to the model's performance, as well as what kinds of semantic relationships are encoded in the resulting phrase embeddings, and whether denotational similarity contains more visual scene information than other entailment models. Finally, we show that compared to standard phrase embeddings, our denotational embedding model learns relationships that express *what happens in a certain type of scene* instead of representing synonym relationships.

7.1 A DENOTATIONAL PHRASE MODEL FOR MPE

In this section, we present a new model to learn phrase embeddings from denotational information and apply these embeddings to the MPE textual entailment task. This model has two main stages. In the first, we train an encoder to predict conditional probabilities for pairs of phrases from the denotation graph. In the second stage, we use the encoder to embed sentences from MPE, and train a textual entailment classifier that uses these representations to outperform standard neural sequence models.

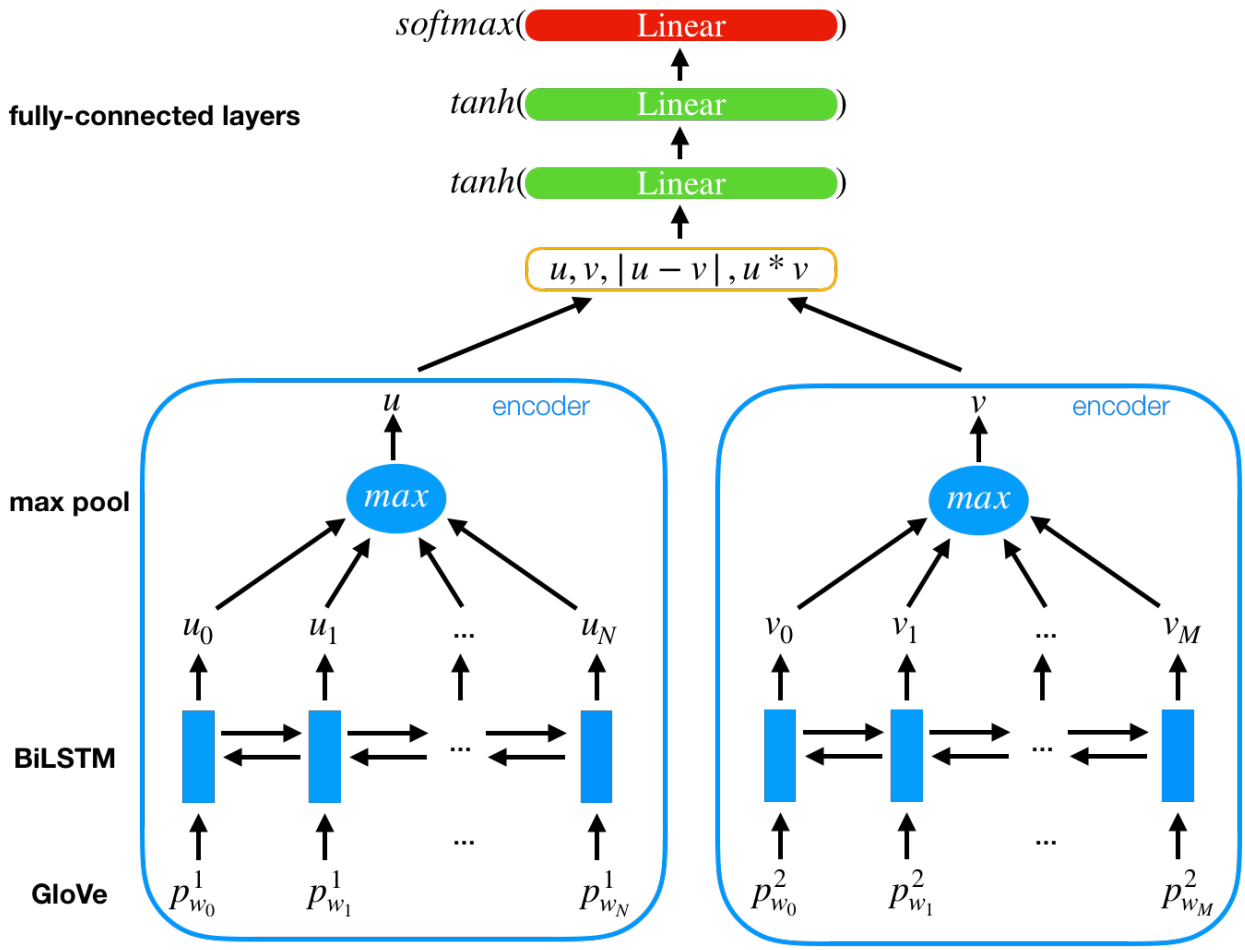


Figure 7.1: The BiLSTM encoder DENENCODE that we train on DENPHRASEBAL to predict denotational probabilities as a five-way classification problem.

7.1.1 Denotational Phrase Encoder

Our denotational phrase encoder, which we will refer to as DENENCODE, is based on INFERSENT [40][1]. It consists of a BiLSTM encoder with max-pooling that produces summary vectors u and v for the two input sentences. We extract features via concatenation (u, v), element-wise multiplication $u * v$, and absolute element-wise difference $|u - v|$. We pass the resulting premise-hypothesis feature vector into a classifier consisting of multiple fully-connected layers and a final softmax layer. We define the classes by discretizing the denotational phrase conditional probability $P_{[\![}(u|v)$ from the DENPHRASE data into five bins. Figure 7.1 illustrates DENENCODE's basic structure.

The bidirectional LSTM takes as input pre-trained 300d fixed GloVe vectors and produces a vector h_t at each step t for word w_t in the sentence. h_t is the concatenation of the output

[1] `https://github.com/facebookresearch/InferSent`

vectors from the forward LSTM and the backward LSTM at time t.

$$\overrightarrow{h_t} = \overrightarrow{LSTM_t}(w_1, ..., w_T)$$
$$\overleftarrow{h_t} = \overleftarrow{LSTM_t}(w_1, ..., w_T)$$
$$h_t = \left[\overrightarrow{h_t}, \overleftarrow{h_t}\right]$$

Finally, the sentence representation u is created by max-pooling (selecting the element-wise maximum value over all vectors) over the word vectors $h_1 ... h_N$. When applying DenEncode to downstream tasks like MPE, we use u and v to represent the sentences, leaving out the extracted features $(u * v, |u - v|)$ that are used to train the encoder initially.

Moving Away from Interpretable Representations

In Chapter 5, we used the predicted conditional probability $\hat{P}_{[\![}(h|d)$ from our previous denotational embedding model DenEmbed as an additional feature in an entailment classifier. This single output value from our model improved the performance of a baseline textual entailment model. However, we failed to train a classifier using the embeddings produced by DenEmbed instead of the single feature $\hat{P}_{[\![}(h|d)$. Even though the embeddings should be more expressive than a single conditional probability prediction, we saw no additional gains from using them.

One possible reason for this result might be the interpretable aspect of our previous embedding model. Because we forced the embeddings to have specific relations in a structured embedding space, we constrained the set of possible vectors that could be produced by the model. This inflexibility could limit the expressivity of the model, so that even when its predicted probabilities are highly correlated with the gold probabilities, the intermediate embeddings do not provide much additional information that is useful to a downstream task like textual entailment.

In contrast, the BiLSTM encoder InferSent, on which DenEncode is based, is trained on a textual entailment dataset like SNLI in order to produce sentence embeddings that are demonstrably useful for a large number of other semantic tasks. While the resulting embeddings no longer exist in a geometrically interpretable space, they are more useful for a downstream textual entailment task. We follow this approach in using a similar encoder architecture for DenEncode; our goal is to train the encoder on denotational phrases and apply it to a downstream entailment task.

Discretizing Probabilities

We previously trained DenEmbed to predict real-value conditional probabilities, but we take a different approach for DenEncode. We take the DenPhrase data and discretize the conditional probabilities into five bins, turning a regression problem into a classification problem. We do this for several reasons. First, the BiLSTM architecture of our encoder was designed for classification tasks and may not produce good results on regression. Furthermore, we are interested not in the exact predicted probability values, but in using the resulting denotational embeddings for a downstream textual entailment task. Therefore, we are necessarily losing important information when we simplify the probability prediction task to a classification task by binning the probabilities into five bins[2]. Previous work, particularly in computer vision, has demonstrated that discretizing a continuous value regression problem in this way can increase prediction accuracy because it simplifies the information space [88, 89, 90, 91, 92].

In discretizing our probabilities, we also subsample the DenPhrase data to ensure approximately equal-sized bins. There are far more phrase pairs with conditional probability $P_{\prod}(x|y) < 0.1$ than in any other bin, so we sample 2% of these pairs for our classification phrase data. Subsampling the phrase data does not eliminate any phrases from the training data, only particular phrase pairs, so the vocabulary size of the training data remains unchanged. We refer to this modified version of DenPhrase as DenPhraseBal.

7.1.2 Entailment Aggregation Classifier

We now use the pre-trained DenEncode to encode MPE sentences, and use those representations to train a classifier to predict entailment. This model, InferDen, is illustrated in Figure 7.2.

To start, we decompose each sentence in MPE into a set of phrases $p_1...p_N$, each of which is passed through the encoder to get a vector representation $e_1...e_N$. We previously observed that since DenPhrase contains short phrases, an encoder trained on this data may not produce reliable representations for longer sentences. Therefore, we decompose each premise sentence into all possible subsequence phrases. We consider subsequences so that we are not limited to phrases that consist of only adjacent words. For example, in the sentence "A boy with a stick kneeling in front of a goalie net," we want to consider phrases like "boy kneeling" and "boy in front of a goalie net" in addition to phrases like "boy with a stick" and "with a stick kneeling." Since this naively generates a huge number of constituent phrases, we first

[2]We also tried three bins with similar results.

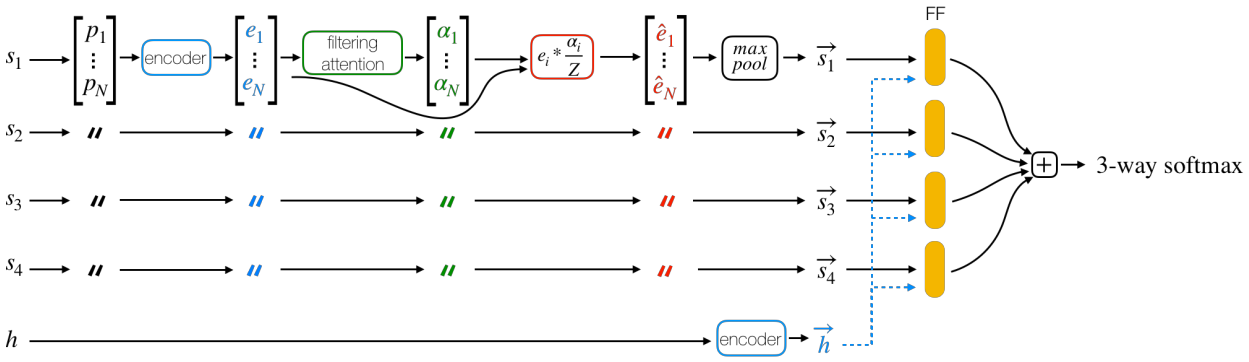


Figure 7.2: The InferDen model that trains an MPE classifier from encoded phrase vectors output by DenEncode.

remove stopwords[3] from the sentence. This process results in an average of 261 constituent phrases per premise sentence.

To combine the set of phrase vectors into a single premise vector representation, we apply attention as a filtering step that can downweight uninformative or ungrammatical phrases, while upweighting phrases that contribute more to the summary sentence vector. We apply the learned attention vector w to each phrase vector e_i in the same sentence to produce unnormalized attention weights. The normalized attention weights α are the result of applying a softmax to the attention weights of all the phrase vectors $e_1 ... e_N$ in the same premise sentence.

$$\alpha_i = \frac{\exp\left(w^T e_i\right)}{\sum_j \exp\left(w^T e_j\right)} \tag{7.1}$$

Then each attention weight α_i is used to reweight the corresponding phrase vector e_i. The reweighted phrase vectors $\hat{e}_1 ... \hat{e}_N$ are summed to produce a single sentence vector s.

$$s = \sum_i \alpha_i * e_i \tag{7.2}$$

Finally, we apply the same prediction aggregation approach from our SumOfExperts model in Chapter 6 to the four sentence vectors. InferDen produces a prediction for each premise vector concatenated with the hypothesis vector by passing the two vectors through a hidden layer, ReLU, and three-class prediction layer. We sum the logits across the four premises and apply a softmax to predict the label of the whole item.

[3]We use the list of English stopwords from NLTK.

7.1.3 Model Variations

In the process of developing the INFERDEN model, we also experimented with multiple alternative model variations. In this section, we describe some of the main modifications that we explored. All of these models produced worse results on the MPE development data compared to INFERDEN, so we do not include a full evaluation on the test data for most of them.

Sentence Decomposition

INFERDEN uses a filtered set of subsequence phrases to represent the premise sentence, but there are alternatives to this decomposition.

Sentence encodings In the simplest approach, $\text{INFERDEN}_{\text{SENT}}$, we do not decompose the premise sentence at all, but pass the entire sentence through the pre-trained encoder to produce a single sentence vector. Unsurprisingly, this approach is not particularly successful: as we observed in Chapter 5, an encoder trained on short denotation phrases does not produce informative embeddings for much longer sentences. We present some results from this model in Section 6.4.

Sliding window substrings Another approach, $\text{INFERDEN}_{\text{SUBSTR}}$, involves considering phrases of adjacent words based on a sliding window. We consider phrases lengths from one word up to k adjacent words, and we do not remove stopwords. This approach produces an average of 47 constituent phrases per premise sentence, much fewer than the 261 average subsequence phrases considered by INFERDEN. We present some results from this model in Section 6.4 as well.

Denotation graph phrases Neither the sliding window phrase approach nor the subsequence phrase approach makes any attempt to filter the set of phrases to contain only informative ones before encoding. Instead, we rely on the model to discount uninformative or superfluous phrases. One possible alternative is to use the denotation graph decomposition of each premise sentence, taking the set of all node phrases corresponding to a sentence as the set of constituent phrases. This should produce only semantically meaningful phrases to the model, and additionally contains hypernym information not included in the other phrase decompositions. However, these phrases are somewhat limited in their coverage due to the brittleness of the sentence simplification algorithm, and some important constituent phrases may be missing. Furthermore, certain parts of the sentence (e.g the subject) tend

to be overrepresented in this phrase set. We do not present any further results from this model.

Combining Phrase Vectors

Given a set of phrase vectors that all come from the same premise sentence, there are many ways to combine these vectors to produce a single sentence vector representation. As we described, InferDen uses the attention as a filtering step to downweight some of the phrase vectors. We pursued several other alternatives to combining phrase vectors, but none of them performed as well as our self-attention approach. We do not present any results from these models.

Hypothesis attention Instead of applying attention weights only to the phrase vectors, this method applies the learned attention vector w to the concatenation $[\cdot, \cdot]$ of each phrase vector e_i and the hypothesis embedding h. We apply softmax to normalize attention weights for each premise.

$$\alpha_i = \frac{\exp\left(w^T [e_i, h]\right)}{\sum_j \exp\left(w^T [e_j, h]\right)} \tag{7.3}$$

Then, as with the filtering-attention model, the attention weight α_i is used to reweight the phrase vector e_i before summing all phrase vectors to produce a single sentence vector $\mathbf{s}$ for each premise.

Max pooling Rather than using attention for reweighting, this approach takes the element-wise maximum over all phrase vectors p_i to produce a single sentence vector s. This approach helped when there were fewer constituent phrases, as in the sliding window decomposition method, but did not produce good results when faced with the much larger set of subsequence phrases.

Combining Premise Vectors

Assuming that the phrase vectors have been combined into a single premise vector, we have four premise vectors that we must combine in some way to make a single prediction for the entailment label of the full item. InferDen makes four separate predictions, one for each premise vector paired with the hypothesis embedding, and sums the resulting logits in order to make the final prediction.

Pooling We can instead max pool or mean pool over the four premise vectors to produce a single summary premise vector which then is concatenated with the hypothesis before passing through a hidden layer and a softmax prediction layer.

Single premise attention We also consider applying attention reweighting to the four premise vectors s_i each concatenated with the hypothesis embedding h.

$$\alpha_i = \frac{\exp\left(w^T\left[s_i, h\right]\right)}{\sum_j \exp\left(w^T\left[s_j, h\right]\right)} \tag{7.4}$$

The premise vector s is the reweighted sum over the four individual premise vectors s_i:

$$s = \sum_i \alpha_i * s_i \tag{7.5}$$

The summary premise vector s is then concatenated with the hypothesis embedding h and passed through a hidden layer and a softmax prediction layer.

Combined premise attention The above methods make no attempt to explicitly combine the vectors of multiple premises, though one premise can affect another premise during training. Since one of the characteristics of MPE is the need to identify whether one premise alone out of the four premises determines the correct entailment label for the full item, or whether information must be aggregated from more than one premise, we consider an approach that considers all possible explicit combinations of premise vectors. We take all combinations of two, three, or four vectors as well as the single premise vectors. We max pool over each combination of vectors to produce a total of 15 summary vectors, each of which we concatenate with the shared hypothesis embedding in order to compute attention weights. The attention weights are computed in the same way as for the single premise attention model before, but there are 15 premise combination vectors instead of only four premise vectors. The final premise summary vector is the sum over the reweighted combination premise vectors, which is concatenated with the hypothesis embedding and passed through a hidden layer and a final softmax prediction layer.

Word Representations

InferDen uses GloVe embeddings to represent words, as these have been the common word embedding representation for most recent textual entailment work. However, we wondered whether GloVe embeddings would be able to account for the vocabulary discrepancy

between the denotation phrase data, which is lemmatized, and MPE, which is not. We ran several experiments to see if there was a different initial word representation that would produce better results on MPE, but none of these alternatives were successful.

Lemmatize MPE One simple potential fix is to continue using GloVe embeddings, but lemmatize MPE so that the vocabulary overlaps more with the denotation phrase data on which the DenEncode encoder is trained.

FastText embeddings We also tried substituting FastText embeddings [93], as they have been trained with subword information and might be better able to recover from OOV issues and share information between the lemmatized and inflected forms of a word. We used the FastText embeddings trained with subword information on 16 billion tokens of English text from Wikipedia 2017, UMBC webbase corpus, and statmt.org news dataset[4]. However, they were ultimately worse than GloVe embeddings for MPE.

GloVe with a nonlinear transformation Finally, we added a nonlinear transformation on top of the GloVe embeddings. This consists of a single layer of the same dimensionality as the GloVe embeddings (300d), followed by a tanh operation. This layer is trained together with DenEncode on DenPhraseBal. We hypothesized that this layer could project the lemmatized word vectors in DenPhraseBal closer to the inflected word vectors in MPE. However, adding this layer did not improve the classification results on MPE.

7.1.4 Parameters

Given a particular model architecture for the classifier, we ran a grid search over a set of parameters: max phrase length in {4, 5, 6}, dropout in {0, 0.5}, hidden layer dimensionality in {128, 256, 512}, batch size in {16, 32}. When pre-training the encoder, we tried reducing the default encoder dimensionality from 4096 to 2048 or 1024, but kept the default values for all other parameters. Table 7.1 contains the parameter values for all our experiments. The best parameters were selected based on a grid search where models were evaluated on the MPE development data.

[4] `https://fasttext.cc/docs/en/english-vectors.html`

Experiment	Model	Hidden dim	Batch size	Dropout	Encoder dim
Ternary	InferSent-MPE	512	32	0.5	4096
	InferSent-SNLI	512	32	0.5	4096
	$\text{InferDen}_{\text{SENT}}$	1024	16	0.2	4096
	$\text{InferDen}_{\text{SUBSTR}}$	512	32	0.0	1024
	$\text{InferDen}_{\text{UG}}$	256	16	0.0	1024
	InferDen	256	16	0.0	1024
Binary	InferSent-SNLI	512	32	0.5	4096
	$\text{InferDen}_{\text{UG}}$	512	32	0.0	1024
	InferDen	512	16	0.0	1024

Table 7.1: The parameters we used to train each model. All models use fixed GloVe vectors as the input word embeddings.

Model	Accuracy
Majority class baseline	42.4
LSTMCond	53.5
InferSent-MPE	53.5
SumOfExperts	56.3
InferDen	**58.0**

Table 7.2: Entailment test accuracy on MPE. LSTMCond, SumOfExperts, and InferSent-MPE are trained only on MPE data. InferDen uses the DenEncode encoder that was pretrained on DenPhraseBal. Adding the denotation data as a pretraining step improves accuracy on MPE.

7.2 EXPERIMENTAL RESULTS

In this section, we compare the performance of InferDen using DenEncode phrase vectors against other neural entailment models on MPE. We show that InferDen produces more accurate entailment predictions than other neural entailment models that were trained on MPE, but does not quite reach the accuracy of models that were pretrained on SNLI. Finally, we present an error analysis comparing InferDen with InferSent-SNLI to see the effects of different types of pretraining data (denotational phrase probabilities compared to labeled entailment sentences).

Model	Accuracy
Majority class baseline	42.4
InferDen	**58.0**
$\text{InferDen}_{\text{SUBSTR}}$	54.4
$\text{InferDen}_{\text{SENT}}$	49.8

Table 7.3: Ablation experiment, showing entailment test accuracy on MPE. InferDen considers all subsequences (contiguous and noncontiguous phrases) up to a certain length. It outperforms $\text{InferDen}_{\text{SUBSTR}}$, which considers only contiguous encoded phrases (substrings), and $\text{InferDen}_{\text{SENT}}$, which does not decompose the premise sentences before encoding.

7.2.1 Classification

In Table 7.2, we compare InferDen (which has been pretrained on denotational phrase data DenPhraseBal and then trained on MPE) to three neural sequence models that are trained on MPE. The models are evaluated in terms of overall accuracy on the MPE test data.

We compare to InferSent-MPE, which is a version of InferSent [40] that we trained only on MPE. To adapt InferSent to the multiple premise setting, we encode each premise sentence separately and predict the entailment label for each premise paired with the hypothesis (similar to our premise-aggregation approach for SumOfExperts). The pairwise predictions are summed and passed to the final softmax for the full label prediction.

We also compare this model to two neural sequence models from Chapter 6: the conditional LSTM model LSTMCond, which treats the four premise sentences as a single long premise sequence, and SumOfExperts, which processes each premise sentence separately and aggregates four pairwise predictions.

Our InferDen model solidly outperforms all of these models; the addition of denotational probability information clearly improves the performance of the entailment model on MPE. This supports our hypothesis that denotational representations capture information that is not readily available to basic neural entailment models. It is encouraging that InferDen produces noticeable gains despite the fact that we had to compensate for the short length of the denotational phrases in pretraining. Further improvements to the denotational pretraining step to address phrase length and vocabulary issues have the potential to produce even better models for downstream textual entailment.

We also present a comparison among variations of our InferDen model that take different approaches to sentence decomposition. In Table 7.3, we compare the accuracy of these models. InferDen considers all subsequences of up to four words in the premise sentences,

Model	Accuracy
Majority class baseline	42.4
InferDen	58.0
InferSent-SNLI	58.9
Attention-SNLI	**64.0**

Table 7.4: Pre-training a comparable model on SNLI first before training on MPE slightly outperforms pre-training on denotation data. These gains are larger for a word-to-word attention model.

both contiguous and non-contiguous phrases. $\text{InferDen}_{\text{SUBSTR}}$ considers only contiguous phrases up to length 4. Finally, $\text{InferDen}_{\text{SENT}}$ passes full premise sentences through the pretrained encoder without decomposing them into shorter phrases. (The details of these models are described in Section 7.1.3.)

All three variations of InferDen use the same underlying phrase encoder, which is pre-trained on DenPhraseBal, achieving 88% accuracy for 5-class classification on the test data. For each model, the pre-trained encoder produces embeddings for the relevant phrases or sentences in MPE. We then train the entailment classifier described in Section 7.1.2 on the fixed phrase/sentence embeddings as input.

The results in Table 7.3 show that there is a clear benefit to looking at noncontiguous phrases over substrings or full sentences. Since the encoder is pretrained on denotational information between short phrases, using it to encode full sentences results in a decrease in accuracy of over 8 percentage points. Conversely, looking at noncontiguous phrases improves InferDen's accuracy by 3.5 percentage points, from 54.4% to 58.0%. Noncontiguous phrases can capture long-distance dependencies in each premise sentence, helping the model aggregate and represent important information within each sentence. Based on these results, one promising avenue of future work could investigate methods to learn the best sentence decomposition approach for the data.

7.2.2 SNLI Pretraining

Finally, we compare InferDen to two models that are pre-trained on SNLI. InferSent-SNLI is the BiLSTM max-pooling encoder model described by Conneau et al. [40]. We pre-train the encoder on SNLI, and then use the trained model to produce sentence vectors for the premises and hypothesis in MPE. We use the encoded sentence vectors to train a classifier that consists of a single feed-forward layer followed by a ReLU to make a separate

prediction for each premise vector concatenated with the hypothesis vector. This layer is trained only on the MPE data. Like the second stage of the InferDen model, we sum over the logits and pass the result through a final softmax to predict the label. Attention-SNLI, the second SNLI-pretrained model, is the word-to-word attention model we presented in Chapter 6.

Table 7.4 shows that InferDen is still outperformed by the two models that are pretrained on SNLI. InferSent-SNLI slightly outperforms InferDen at 58.9%, while the word-to-word attention model continues to be the best performing model overall at 64.0%. Despite the gains that we see from adding denotational information to MPE models, it is still not as effective as adding more labeled information in the form of SNLI sentence pairs. Even though MPE is not a sentence pair task, there is still a lot of information about entailment patterns that can be transferred from SNLI to MPE. To some extent, this is not surprising: SNLI contains a lot of precisely labeled data, in comparison to the noisy automatic process by which we compute conditional probabilities from the denotation graph.

In sum, our experiments show that InferDen demonstrates a clear improvement over neural entailment models trained on MPE, including our previous SumOfExperts model, but does not quite reach the performance of a similar model that uses SNLI data. In the rest of this chapter, we will explore the differences between denotational- and SNLI-pretrained models, and examine how denotational embedding models compare to other kinds of phrase similarity.

7.2.3 Error Analysis

In this section, we present an error analysis comparison InferDen and InferSent-SNLI. We chose InferSent-SNLI rather than Attention-SNLI because it is similar in design to InferDen, and presents a fairer comparison of the effects of the different pretraining data (SNLI vs. DenPhraseBal).

We first identify the set of errors that each model makes consistently. We ran each model 10 times and collected their predictions on the development data. For this analysis, we looked only at the items that each model consistently labeled incorrectly for all 10 runs. We look for any patterns in the types of errors consistently made by InferSent-SNLI but not InferDen, and vice-versa. We provide some examples of each error type for discussion purposes (with the gold label indicated by $\Rightarrow$).

Overall, both models have the most difficulty with neutral items. This is not surprising, as it is also the category which had the lowest agreement among our crowdsourced labels. Among the errors consistently made by InferSent-SNLI were some examples of

fairly straightforward synonym replacement. These synonyms (⟨*converse*, *talk*⟩, ⟨*talk*, *have a discussion*⟩, ⟨*jump*, *leap in the air*⟩) are relatively common co-occurrences in FLICKR30K and therefore may have reasonable support in the denotational embeddings, but may be less common in the human-authored hypotheses in SNLI.

PREMISES:
Three women standing in front of a Sallie Mae sign interact among themselves.
Some women wearing dark clothing are talking in front of a Sallie Mae sign.
A group of women standing in front of a blue Sallie Mae banner.
Three females talking in front of a Sallie Mae banner.

HYPOTHESIS: Older women converse. ⇒ENTAILMENT

As for NEUTRAL errors that are consistently made by INFERSENT-SNLI but not INFERDEN, INFERDEN's correct predictions may be due to the denotation graph's weak but nonzero support for the co-occurrence of some phrases that sometimes describe the same scene. Given this information – that the events co-occur sometimes but are not guaranteed to do so – the denotational model correctly classifies these cases as NEUTRAL, while INFERSENT-SNLI makes errors. In the following example, a man selling magazines may or may not be sitting. The NEUTRAL label is a reasonable prediction because the denotation graph contains some instances of *a person selling something* where the person is sitting down, and some instances where the person is standing.

PREMISES:
An older man sells magazines at a magazine stand in Asia.
A magazine stand with many multicolored magazines.
An elderly man sells magazines in a stand.
An elderly man selling magazines.

HYPOTHESIS: An old man sitting. ⇒NEUTRAL

Finally, INFERSENT-SNLI tends to make CONTRADICTION errors that INFERDEN avoids in cases where the hypothesis is too specific and contains some detail that is unlikely to be true of the scene described by the premises. In this example, the hypothesis *"A man playing his instrument"* would be entailed by the premises, but the addition of the scene information *on a corner* makes the hypothesis much less likely given the premise description of the scene as a *concert*.

PREMISES:
A man in a brownish green shirting playing guitar singing into a microphone.
A man is playing guitar and singing into a microphone.
A man playing the guitar and singing.
A young man singing at a concert.

HYPOTHESIS: A man playing his instrument on a corner. ⇒CONTRADICTION

In the other direction, we can look at the errors consistently made by INFERDEN but not INFERSENT-SNLI. One common error involving NEUTRAL items is when the premises describe a scene that either explicitly or implicitly contains people other than the main subjects mentioned, and the hypothesis describes an entity that could conceivably exist in the background of that scene. This is common with crowd scenes: if the premises mention a crowd of people in the background, then a large number of hypotheses could possibly describe some person in the crowd. In the example below, although none of the premises specifically mention *men sitting at a table*, two of the premises mention people working somewhere in the scene who could easily be the men in the hypothesis.

PREMISES:
A woman is sitting at a desk on her computer, and other people are behind her with laptops in front of them as well.
A woman on the side of the picture typing on a keyboard with a laptop computer in front of her.
Many people busily working, watching a presentation, or talking to each other.
A woman in a pink v neck shirt is typing on a keyboard plugged into a laptop.

HYPOTHESIS: Men sitting behind a table. ⇒NEUTRAL

Other NEUTRAL errors involve strong world knowledge about what kinds of actions are compatible with one another (but not strongly implied or required to be true given the premises). This requires a stronger signal about compatible or mutual exclusive events than the denotation graph has: in its current form, it is difficult to distinguish events that are unseen together because they are incompatible from events that are unseen together simply by chance in the data we collected. In the following example, workers can definitely be *smiling* while doing their jobs with no contradiction, but it is unlikely that we observed this co-occurrence in the denotation graph with sufficient frequency to correctly make that prediction.

PREMISES:
Two men in hard hats, business suits, and orange vests walking through a facility's yard.
The two men in suits and safety gear are overseeing construction.
Two men with construction hats walk next to concrete structure.
Two men inspecting and building a wall.

HYPOTHESIS: Workers smile. ⇒NEUTRAL

Many of the ENTAILMENT errors made only by INFERDEN involve some implications that are relatively easy for humans to make given some amount of world knowledge, but are perhaps difficult to distill to simple co-occurrences among denotation graph phrases. For example, *people posing for a photo* are likely to be *smiling*.

PREMISES:
A sports team made up of 14 women wearing white t-shirt and pink tags.
A group of runners and walkers posing together for a picture.
A group of women posing for a picture.
Cancer survivors race for a cure.

HYPOTHESIS: Women smiling. ⇒ENTAILMENT

The CONTRADICTION errors made by INFERDEN occur in cases where the hypothesis describes an event that is theoretically possible in the premise scene, but extremely unlikely. INFERSENT-SNLI may have more success with these cases because it has the advantage of many more labeled negative examples than the denotation graph and INFERDEN.

PREMISES:
A football player wearing a number 23 white jersey tackles a football player wearing a red, number three jersey while fans cheer.
A football player being tackled from behind by another player from a different team in front of a crowd.
A football tackle in progress with a crowd in the background.
A football player in red being tackled by opponents in white.

HYPOTHESIS: A dog running through a snowy field. ⇒CONTRADICTION

7.3 DENOTATIONAL EMBEDDING ANALYSIS

One area of exploration that still remains is to examine what kind of information the denotational embedding model is capturing. In Chapter 3, we presented a brief initial exploration

of this question in the context of the gold denotational probabilities, but we have not performed a similar examination of the representation produced by a denotational embedding model. In this section, we present several experiments to break down the sources of denotational information and determine what types of relationships are captured in denotational embedding models like InferDen.

7.3.1 The Effects of Grounding on the Denotation Graph

The motivation for constructing the denotation graph and the succeeding models was to learn a representation of language that is grounded in some other depiction of the world. In our implementation, we chose images to be this grounding. With this additional knowledge of the grounding of linguistic expressions – specifically, knowing that two captions both describe the same image – we predicted that the resulting denotational similarity would capture semantic relationships that are different from those captured by distributional similarity and specifically useful for tasks like textual entailment. This aspect of the denotation graph and denotational similarity – the fact that we have multiple captions grounded in a common image – is somewhat unique to image caption datasets. Replicating this in another domain would be a nontrivial process.

However, there is another key aspect of constructing the denotation graph that also improves our model's coverage on downstream tasks: the generation process that we use to produce more general descriptions of each image from the original captions. This relies on a small set of hand-defined rules that each produce a new sentence that is more general but still true given the original sentence. This process increases the coverage of the denotation graph, producing more phrases and identifying more linguistic expressions that share a common image denotation. This part of the graph construction process could be applied to any other text corpus.

In thinking about whether we can generalize the denotational approach to other data sources and domains, an important question is how much the initial grounding to common images contributes to the ultimate performance. We have hypothesized that denotational similarity is a useful metric because of the image grounding step, but we want to be able to quantify this claim. To do this, we recomputed the gold probabilities in the DenPhraseBal dataset as if each caption described its own unique image, and used this new dataset ($\text{DenPhraseBal}_{\text{UG}}$) to train an ungrounded InferDen model.

To compute the gold probabilities in DenPhraseBal, we used Equations 3.2–3.4, which compute the probability of a phrase x as the number of images described by x and the joint probability of phrases x and y as the number of common images that they both describe in

Model	Accuracy
InferDen$_{\text{SENT}}$	49.8
InferDen$_{\text{SUBSTR}}$	54.4
InferDen	**58.0**
InferDen$_{\text{UG}}$	53.8

Table 7.5: Comparing grounded InferDen models to an ungrounded model using entailment accuracy on MPE test.

the corpus. To re-compute these probabilities for a denotation graph over a set of captions that each describe a unique image, we redefine the denotation function $[\![\cdot]\!]$ to map a sentence s to the set of *captions* $c \in U$ that contain s, rather than the set of images that s describes. We then recompute Equations 3.2–3.4 given this new definition of denotation, producing new gold probabilities for an ungrounded denotation graph.

We train another InferDen model, InferDen$_{\text{UG}}$, on the new DenPhraseBal$_{\text{UG}}$ dataset of phrase probabilities, and use the resulting phrase embeddings on subsequences of up to 4 words to train a classifier on MPE. In Table 7.5, we compare the results of InferDen$_{\text{UG}}$ to the entailment classification results of the grounded InferDen models from Section 7.2. As we expected, the ungrounded model is much worse than the grounded models, performing more than 4 points lower than the grounded model with the same configuration (InferDen).

This result is not surprising, since we expected that having no common images between captions would mean that we miss out on valid co-occurrences between phrases and capture less linguistic variation overall. Nevertheless, this experiment is a helpful quantification of the significance of this part of the data collection process, and tells us that we would want to follow a similar image- or scene-grounded approach in a new domain.

Error Analysis

We can also examine how the errors made by InferDen$_{\text{UG}}$ differ from the grounded InferDen. As in Section 7.2.3, we run each model ten times and look for patterns in the errors that were consistently made by each model.

One key difference is that InferDen appears to have better coverage of paraphrases. InferDen$_{\text{UG}}$ misses many examples of entailment where the hypothesis is synonymous with some part of the premises, while InferDen correctly identifies that ⟨*man kayaks*, *man paddles*⟩, ⟨*infant screams*, *child yells*⟩, and ⟨*talking*, *conversing*⟩ are paraphrases and therefore the hypothesis is entailed.

Another distinction is that INFERDEN does better on items where the hypothesis states something about the scene that would be obvious from the image being described. These are often unstated implications: for example, that *a dog shaking off water* is *standing*, and that *a woman playing the violin* is *holding an instrument*. As INFERDEN has better coverage of linguistic variation and implications for a shared scene, it makes sense that INFERDEN does better than INFERDEN$_{\text{UG}}$ on these implications as well as paraphrases and synonyms.

7.3.2 MPE as an Image-Grounded Binary Evaluation

In comparing INFERDEN$_{\text{UG}}$ to INFERDEN, we wanted to investigate to what extent the presence of images, which capture rich scenes and can be described by a wide variety of different descriptive captions, affects our denotational representation. We present an evaluation to quantify what inferences the grounded INFERDEN model can make given that it was indirectly informed by images.

This evaluation is similar to the ATE experiment that we described in Chapter 3. Like ATE, this is a binary classification task where each premise-hypothesis item has been labeled according to whether the hypothesis describes the same image as the premises. We relabel the MPE data with binary labels based on how the items were generated: roughly 50% of the items have a hypothesis that describes the same image as the premises. We label these items as ENTAILMENT because the hypothesis is entailed by the image. The remaining 50% of the items have a hypothesis from a different image; we labels these items as CONTRADICTION because the hypothesis contradicts the image. Although the new label depends on the image, our models are text-based so they are presented with only the text of the premises, which present an impoverished version of the scene information available in the image. This means that some of the new binary labels may appear to be inconsistent with the premise text, because they depend on information present only in the image. As a result, this task evaluates whether these models capture information about typical visual scenes that can be inferred based on incomplete information in the premise text.

This grounded version of MPE is conceptually related to the grounded textual entailment task that Vu et al. [94] explore. However, we are evaluating text-based models only, while Vu et al. investigated multimodal (vision and language) models. Unlike SNLI, MPE actually has ground truth labels as to whether the hypothesis is entailed or not by the original image, allowing us to evaluate whether these models can accurately capture the relationship between the hypothesis, the premises, and the (unseen) image.

Figure 7.3 shows an example of the binary image-based relabeling process, illustrating how including the image disambiguates previously NEUTRAL instances. The four premises on the

Hypothesis: A vendor sitting.

Premises:

A man is laying out his merchandise to include eggs, chickens and vegetables.
A man in a brown coat and pants is arranging dead chickens on a mat.
A man sells eggs and chickens from a mat on the street.
A man with dead chickens and baskets of chicken eggs.

Image:

$\Longrightarrow$ NEUTRAL $\Longrightarrow$ ENTAILMENT

Figure 7.3: Left: an example from MPE where the truth of the hypothesis is ambiguous (NEUTRAL) given the premises. Right: the label is resolved as ENTAILMENT given the image that the premises describe.

Model	Accuracy All	Neutral
Majority class baseline	50.2	42.1
InferSent-SNLI	**69.5**	62.7
InferDen$_{UG}$	66.8	63.4
InferDen	68.0	**64.5**

Table 7.6: Entailment test accuracy on binary grounded MPE task. While InferSent-SNLI has the highest overall accuracy, InferDen has the highest accuracy on the subset of test data where the image is required to disambiguate the label from NEUTRAL to ENTAILMENT or CONTRADICTION.

left describe a scene where a man is setting up his merchandise to sell on the street. There is not enough information in the premises to tell whether the hypothesis, which states that the vender is sitting, is true or false. Therefore, the label of this item in MPE is NEUTRAL. However, if we look at the image that the premises were written to describe (right side), it is clear that the man is sitting down and therefore the hypothesis is entailed. The goal of this task is to evaluate whether models can predict the right-hand label, which is conditioned on the image, given only the premise text on the left.

We train models on binary-labeled MPE, and present the test results in Table 7.6. We compare InferDen to a corresponding ungrounded model, which again performs worse than InferDen on this binary task, and to InferSent-SNLI, which continues to slightly outperform InferDen, achieving 69.5% accuracy compared to InferDen's 68.0% accuracy

on the test data.

However, the overall test accuracy includes items that were relabeled as well as items whose label did not change from the ternary text-based task to the binary image-based task. We already know that InferSent-SNLI has higher accuracy than InferDen on the ternary task. For the purposes of this analysis, we are actually interested in the subset of the MPE data where the item was labeled neutral in the ternary task and was disambiguated to either entailment or contradiction for the binary image-based task. In the right-hand column of Table 7.6, we compare the accuracies of these models only on the ambiguous neutral items, and find that InferDen is in fact better than InferSent-SNLI at predicting the image-based labels. This confirms our intuition that denotation captures visual inferences that are not apparent from text, but can be verified from the accompanying image. Since none of the models see the image, the fact that InferDen outperforms InferSent-SNLI on the neutral items means that it contains extra-linguistic information about the types of scenes that are tend to accompany the provided premise text.

Error Analysis

We compare InferDen and InferSent-SNLI's errors on the binary grounded task, using the errors made by each model consistently over 10 repeated runs. InferSent-SNLI tends to make incorrect predictions on contradiction items where the hypothesis describes an action that is unusual enough given the premise scene that it would have been mentioned if it were true. In the following example, which InferDen labeled correctly, *digging* is a salient action in these scene that should have been mentioned in the premises; since it wasn't, the hypothesis is not entailed by the premises.

Premises:
Two men in hard hats, business suits, and orange vests walking through a facility's yard.
The two men in suits and safety gear are overseeing construction.
Two men with construction hats walk next to concrete structure.
Two workers are walking around a construction site.

Hypothesis: Men digging. ⇒contradiction

InferSent-SNLI also tends to make mistakes on entailment items where the hypothesis is not really a paraphrase, and may not even be strictly entailed by the premises, but is compatible with the scene. InferDen has better coverage of co-occurrences grounded in common images, so it is better able to extrapolate when the hypothesis is entailed not by the premise text but by the imagined scene about which the premises were written. In

the following example, none of the premises mention *fans watching*, but it is reasonable to assume that a hockey game has an audience. InferDen reasonably labels this item as entailment, which is the correct label given the image context.

Premises:
A red-uniformed hockey player is attempting to control the puck while two white-suited hockey players try to disrupt him.
A hockey player wearing a red uniform reaches for the puck as others follow.
A hockey player in white tries to steal the puck from another player in red.
People playing hockey on ice.

Hypothesis: Fans watching. ⇒entailment

7.3.3 Comparing Denotational Embeddings to Other Embedding Models

In this final analysis, we step away from downstream evaluations and look at how the representation learned by the denotational encoder, DenEncode, differs from representations learned by distributional word models or other neural encoding models. To do this, we compare the nearest neighbors of a set of target phrases according to three similarity metrics: denotational conditional probabilities (both gold probabilities computed from the denotation graph as well as predicted probabilities from DenEncode), the cosine similarity of vectors composed from GloVe word embeddings, and the cosine similarity of vectors produced by the encoder from InferSent-SNLI.

Data

Most intrinsic embedding analyses look primarily at the similarity between pairs of words. However, since the denotation graph produces similarities over multi-word phrases, we want to look at the nearest neighbors of phrases as well as single words. Therefore, we define our vocabulary using phrases from the denotation graph, where we expect all the models to have reasonable coverage.

We defined three nearest neighbors experiments. In each experiment, for each model and each phrase in the vocabulary, we compute the nearest phrases from the vocabulary according to that model's similarity metric. We analyze how these neighborhoods differ between models.

Observed phrase pairs In the first experiment, the vocabulary consists of all phrases present in the training or development data of DenPhraseBal. Any phrase in the vocabulary can be a neighbor of any other phrase, with no restrictions. We expect that DenEncode will predict the denotational conditional probability of these phrase pairs with reasonably high accuracy, as they were observed in training or validation.

Observed phrase pairs, filtered As we will show, the neighborhoods generated by observed phrase pairs tend to contain mostly phrases with high word overlap with the target phrase (for all models). In the second experiment, we restrict the phrases that can appear in the neighborhood of the target phrase to look for other patterns. If the potential neighbor phrase n is a subset of the target phrase t, or vice-versa, then we remove n as a potential neighbor (essentially forcing $\text{sim}(n, p) = 0$). In addition, specific to denotational similarity, we remove n if $P_{[\![]\!]}(t|n)$ or $\hat{P}_{[\![]\!]}(t|n)$ is equal to 1.0, to ignore ancestor-descendant phrase pairs.

Random phrase pairs In the third experiment, we look at phrase pairs that were not necessarily observed in training DenEncode, to see how well the denotational embedding model generalizes to infrequent, unseen events. The vocabulary consists 10,000 phrases, randomly sampled from the denotation graph. Most phrase pairs in this random sample do not have a common image according to the denotation graph.

Models

Denotational conditional probability Although most semantic similarity metrics are symmetric, textual entailment is a directed relationship. We have therefore found it most useful to focus on the directed denotational conditional probability of one phrase given another. Although it is uncommon to compare symmetric and asymmetric similarities, this comparison should nevertheless provide an interesting look at what semantic relationships are prioritized by the denotational model compared to standard representations like GloVe embeddings.

We use both the gold probabilities according to the denotation graph and the predicted probabilities according to DenEncode. As the similarity metric, we consider both $\hat{P}_{[\![]\!]}(t|n)$, the conditional probability of the target phrase t given a potential neighborhood phrase n, as well as $P_{[\![]\!]}(n|t)$. The gold probabilities come directly from the graph: we compute Equation 3.4 according to the denotation graph (by our standard threshold, this value is 0 if the phrases have fewer than 10 images in common). We refer to the denotation graph probabilities $P_{[\![]\!]}(t|n)$ as $P^1_{[\![]\!]}$ and $P_{[\![]\!]}(n|t)$ as $P^2_{[\![]\!]}$. For predicted probabilities, we use a variation

of DENENCODE which has been trained on DENPHRASEBAL to predict the exact probability value of each phrase pair (rather than the simplified five-way classification model we used in Section 7.1). We refer to the predicted denotational encoder probabilities $\hat{P}_{[\![]\!]}(t|n)$ as $\hat{P}^1_{[\![]\!]}$ and $\hat{P}_{[\![]\!]}(n|t)$ as $\hat{P}^2_{[\![]\!]}$.

GloVe We start with GloVe word vectors (300d, 840B tokens). We compute phrase vectors to be the mean of all the word vectors in the phrase. The similarity metric for a pair of phrases is the cosine similarity of the composed phrase vectors. We refer to these phrase similarities as GLOVE.

InferSent-SNLI We use the encoder from the INFERSENT-SNLI model to produce an output vector for each phrase in the vocabulary. The similarity metric for a pair of phrases in the vocabulary is the cosine similarity between their phrase vectors as produced by the encoder. We refer to these phrase similarities as SNLI for the rest of this section.

Results

Observed phrase pairs We compare the lists of nearest neighbors according to the gold and predicted denotational probabilities and the non-denotational models, GLOVE and SNLI. Table 7.7 contains some examples. We observed that the nearest neighbors of each target phrase are almost always phrases that have very high word overlap with the target phrase. For GLOVE, this is presumably because our method of composing phrase vectors from GloVe word vectors biases the phrase vectors to be close to the constituent word vectors. For $P^1_{[\![]\!]}$ and $\hat{P}^1_{[\![]\!]}$, the nearest neighbors of the target phrase are parent phrases of the target phrase, meaning that they entail the target phrase (and that $P_{[\![]\!]}(t|n)$ or $\hat{P}_{[\![]\!]}(t|n)$ equals 1.0). The target phrase is often a subset of the words of the parent phrase due to the denotation graph construction process: each edge is defined by a lexical transformation like dropping modifiers or prepositional phrases. (Replacing a word with its hypernym does result in a lexical difference, but it only modifies a single word.) For SNLI, we presume that the high word overlap is because the GloVe word vectors used to initialize the model still play a significant role in determining the phrase vector.

Compared to the other models, $P^2_{[\![]\!]}$ and $\hat{P}^2_{[\![]\!]}$ produce fewer phrases with high word overlap with the target phrase. This is because these models select for neighbor phrases that are highly likely given the target phrase, and therefore these phrases are often hypernyms (*adult* or *person* from *blond woman*) or very general descriptors associated with the same scene as

the target phrase (the subject of *play* is almost always a *person* who may be wearing specific *clothing*).

We observe that DenEncode's predicted denotational probabilities are reasonably reliable compared to the gold probabilities from the denotation graph: the nearest neighbor phrases based on predicted probability are similar to the nearest neighbor phrases based on gold probability. For both gold and predicted denotational probabilities $P^1_{[\![]\!]}$ and $\hat{P}^1_{[\![]\!]}$, the closest phrases to *blond woman* describe a blond woman doing something, and the closest phrases to *play* describe a person playing an instrument or a game. These results reinforce our conclusion that DenEncode indeed produces phrase embeddings that capture the denotational conditional probability of phrase pairs.

Observed phrase pairs, filtered We considered the unfiltered neighborhoods in the previous experiment for completeness, but it is more interesting to compare the neighborhoods after removing phrases with trivial relationships to the target phrase. Table 7.8 contains examples of these neighborhoods.

DenEncode continues to produce accurate denotational probabilities and the predicted denotational nearest neighbors hold the same kind of relationships to the target phrase as the gold denotational nearest neighbors. $P^2_{[\![]\!]}$, $\hat{P}^2_{[\![]\!]}$, GloVe, and SNLI continue to group short phrases closer to the target phrases, while $P^1_{[\![]\!]}$ and $\hat{P}^1_{[\![]\!]}$ tend to select longer phrases as nearby neighbors. $P^2_{[\![]\!]}$and $\hat{P}^2_{[\![]\!]}$ tend to select more general phrases like *person*, *adult*, or *clothing*.

However, unlike the unfiltered observed phrase pairs, this filter reveals that denotational similarities $P^1_{[\![]\!]}$ and $\hat{P}^1_{[\![]\!]}$ produce nearest neighbors that represent different semantic relationships than non-denotational similarity. The non-denotational models, GloVe and SNLI, identify phrases that are distributionally similar to the target phrase, which means the neighboring phrases are often synonyms of the target phrase (*accept* $\rightarrow$ {*have, receive, hold*}, *city street* $\rightarrow$ {*city sidewalk, city road, city intersection*} or short phrases involving a synonym of the target phrase (*machine* $\rightarrow$ {*use equipment, with machinery*}. In contrast, both $P^1_{[\![]\!]}$ and $\hat{P}^1_{[\![]\!]}$ tend to select for neighbors that describe the same scene as the target phrase, but are not synonyms. $P^2_{[\![]\!]}$ and $\hat{P}^2_{[\![]\!]}$ include some of these scene-related phrases to a lesser extent (*award* as a neighbor of *accept* from $P^2_{[\![]\!]}$ and *city sidewalk* as a neighbor of *city street* from $\hat{P}^2_{[\![]\!]}$). However, for the most part, the denotational similarities $P^2_{[\![]\!]}$ and $\hat{P}^2_{[\![]\!]}$ result in neighbor phrases are extremely general and less interesting than $P^1_{[\![]\!]}$ and $\hat{P}^1_{[\![]\!]}$, so we will focus our remaining discussion mostly on $P^1_{[\![]\!]}$ and $\hat{P}^1_{[\![]\!]}$.

$P^1_{[\![]\!]}$ and $\hat{P}^1_{[\![]\!]}$ produce neighbors for *accept* that consist of *things that are accepted* (*award*, *trophy*, *something (from someone)*) and a few phrases that describe a scene where *a person accepts something like an award* (an action like *man shake hand* or a recipient like *race*

	Model	Nearest Neighbors
blond woman	$P^1_{[\![]\!]}$	blond woman walk down sidewalk, blond woman wear clothing and glasses, blond woman wear uniform, blond woman wear yellow clothing, smile blond woman
	$P^2_{[\![]\!]}$	woman, blond adult, adult, person, blond person
	$\hat{P}^1_{[\![]\!]}$	blond woman with shirt and pant hold, blond woman wear clothing sing into microphone with adult, blond woman wear shirt sing into microphone with person, blond woman with clothing and pant hold, blond woman wear shirt sing into microphone with adult
	$\hat{P}^2_{[\![]\!]}$	person, adult, clothing, man, blond adult
	GloVe	young blond woman, blond woman dressed, blond woman look, blond woman put, with blond woman
	SNLI	blond lady, with blond woman, young blond woman, blond woman be, two blond woman
play	$P^1_{[\![]\!]}$	crowd of person play, asian person play stringed instrument, adult with glasses play brass instrument, young child play game, some adult play hockey
	$P^2_{[\![]\!]}$	person, person play, adult, clothing, wear clothing
	$\hat{P}^1_{[\![]\!]}$	male person play game while person stand, person wear black clothing play guitar and adult play stringed instrument, woman wear black and red play violin, person wear shirt play stringed instrument while adult play percussion instrument, person sing into microphone while adult play in background
	$\hat{P}^2_{[\![]\!]}$	adult, person play, play instrument, ball, man
	GloVe	play game, game play, play player, player play, player play game
	SNLI	one play, both play, that play, play one, another play
air	$P^1_{[\![]\!]}$	man jump in air on skateboard, rider jump in air, girl into air, girl wear clothing jump into air, dog jump in air catch frisbee
	$P^2_{[\![]\!]}$	person, in air, adult, man, jump
	$\hat{P}^1_{[\![]\!]}$	white animal jump in air catch frisbee, white dog jump in air catch frisbee, dog leap into air catch ball in mouth, brown dog leap into air catch ball, person jump in air with arm and leg
	$\hat{P}^2_{[\![]\!]}$	adult, into air, person, jump in air, person wear clothing
	GloVe	with air, into air, cold air, some air, get air
	SNLI	with air, into air, through air, open air, on air
store	$P^1_{[\![]\!]}$	store display window, clean store window, in front of store window, into store window, stand in front of store window
	$P^2_{[\![]\!]}$	person, adult, shop, man, clothing
	$\hat{P}^1_{[\![]\!]}$	two adult walk down street with shop bag, two woman walk down street with shop bag, person wear yellow clothing stand outside store, woman walk carry two shop bag, man wear clothing and pant stand in front of store
	$\hat{P}^2_{[\![]\!]}$	person store, person, in store, person wear clothing, adult
	GloVe	store in store, grocery store, retail store, warehouse store, shop in grocery store
	SNLI	shop, be store, behind store, above store, with store

Table 7.7: Nearest neighbors of observed phrase pairs.

	Model	Nearest Neighbors
accept	$P^1_{[\![]\!]}$	award, winner, something from person, something from adult, adult hand person, person hand person, trophy, man receive, man shake hand, hand person
	$P^2_{[\![]\!]}$	adult, man, wear clothing, clothing, person wear clothing, adult wear clothing, man wear clothing, stand, woman, award
	$\hat{P}^1_{[\![]\!]}$	something from adult, man receive, something from person, person wear yellow jumpsuit, person hand person, man shake hand, race car driver, race car adult
	$\hat{P}^2_{[\![]\!]}$	man, person, clothing, wear clothing, two, child, shirt, group, wear shirt
	GloVe	person receive, person hold, receive, man receive, person stand, have, hold
	SNLI	have, that, receive, hold, someone, who, one, something, present, money
city street	$P^1_{[\![]\!]}$	down busy sidewalk, walk down busy sidewalk, woman and person walk down street, crosswalk in city, through crowd street
	$P^2_{[\![]\!]}$	adult, on street, man, clothing, wear clothing
	$\hat{P}^1_{[\![]\!]}$	city sidewalk in front of store, woman stand on city sidewalk, man walk on city sidewalk, city sidewalk at night, woman walk on city sidewalk
	$\hat{P}^2_{[\![]\!]}$	person, adult, city sidewalk, man, person wear clothing
	GloVe	city sidewalk, city road, sidewalk in city, city corner, down city sidewalk
	SNLI	city road, city sidewalk, city corner, city intersection, near street
machine	$P^1_{[\![]\!]}$	atm, some machinery, operate machinery, arcade game, casino, machinery, dryer, construction vehicle, in laundromat, make beverage
	$P^2_{[\![]\!]}$	adult, man, clothing, wear clothing, person wear clothing, adult wear clothing, work, shirt, man wear clothing, wear shirt
	$\hat{P}^1_{[\![]\!]}$	with machinery, at loom, adult wear hard hat work, person wear hard hat work, woman operate, person do job, use equipment, worker cut
	$\hat{P}^2_{[\![]\!]}$	person, person wear clothing, adult, adult wear clothing, wear clothing, person wear shirt, clothing, man, wear clothing, man wear shirt
	GloVe	with machinery, operate machinery, some machinery, heavy machinery, work with machinery, use equipment, equipment, work on equipment, system
	SNLI	machinery, equipment, system, steam, contraption, operate, with machinery, use
auditorium	$P^1_{[\![]\!]}$	lecture hall, adult sit on stage, person sit on stage, adult give lecture, two adult give, person give lecture, sit on stage, group sit in room
	$P^2_{[\![]\!]}$	adult, man, group, group of person, person sit, woman, clothing, wear clothing, stage, person wear clothing
	$\hat{P}^1_{[\![]\!]}$	lecture hall, give lecture, person give lecture, adult give lecture, group sit in room, group of person sit in room, in hall, entertainer rehearse, adult sit on stage, adult with instrument sing
	$\hat{P}^2_{[\![]\!]}$	man, clothing, wear clothing, hold, two, shirt, person wear clothing, stand, talk, crowd
	GloVe	lecture hall, hall, in hall, room, lecture, concert, crowd, in room, at podium
	SNLI	hall, in hall, room, lecture hall, lecture, meeting, crowd, podium, stage, front

Table 7.8: Nearest neighbors of observed phrase pairs, with neighbors filtered by word overlap and gold denotational probability.

car driver). SNLI neighbors contain a few *things that are accepted*, but largely consist of synonyms for *accept*.

The nearest neighbors for denotational models $P^1_{[\![]\!]}$ and $\hat{P}^1_{[\![]\!]}$ for the phrase *city street* are not so dramatically different from the non-denotational models: these phrases tend to describe what is happening on a city street (*walk down busy sidewalk*) or other elements of a *city street* scene (*woman stand on city sidewalk*). For the target phrase *machine*, $P^1_{[\![]\!]}$ selects types of machines (*atm, dryer, construction vehicle*), and $\hat{P}^1_{[\![]\!]}$ selects phrases about people who are working with machines (*adult wear hard hat work, woman operate, worker cut*).

In some cases, $\hat{P}^1_{[\![]\!]}$ does identify some multi-word synonyms that are not captured well by the non-denotational model. For the target phrase *adult walk up stair*, both $P^1_{[\![]\!]}$ and $\hat{P}^1_{[\![]\!]}$ include phrases containing *climb* and *ascend* among their twenty nearest neighbors, while the non-denotational models only find *climb* as a close synonym.

Not all target phrases produce such different results between the denotational and non-denotational models. One example is *auditorium*, where all models' neighbor lists contain synonyms (*hall*, *lecture hall*) as well as examples of *people or objects in an auditorium* (*crowd*, *podium*) and *events that happen in an auditorium* (*give lecture*, *entertainer rehearse*, *meeting*).

Random phrase pairs We have shown that DenEncode produces probabilities that capture different semantic relationships from GloVe or SNLI. This observation is based on phrase pairs that occurred in the training or development split of DenPhraseBal. In other words, the phrase pairs whose similarity we have examined all co-occurred with a shared image in the Flickr30K denotation graph.

We can divide the space of all possible event pairs into three categories with regard to some corpus: positive labeled pairs (events that were definitely observed to co-occur), negative labeled pairs (events that were definitely observed not to co-occur), and unobserved pairs (events that were not observed together given some data, but may or may not co-occur in reality). With regard to the denotation graph and its images, we have positive labeled pairs: the co-occurring events that pass our threshold, from which we can compute denotational probabilities. The graph does not have explicitly labeled negative events. However, to address this, we added phrase pairs $\langle x, y \rangle$ *s.t.* $P_{[\![]\!]}(x, y) = 0$ to DenPhrase (and DenPhraseBal) where x and y occurred with sufficient frequency that given independence assumptions, we would have expected them to share at least one image. These two types of events make up the data that we have used to evaluate both our denotational embedding models in Chapter 5 and this chapter. However, we have not investigated how the probabilities produced by DenEncode generalize to events that were not observed with sufficient

frequency in the denotation graph for us to label them with certainty.

To this end, we selected 10,000 random phrase pairs from the denotation graph phrases. Although each individual phrase comes from the denotation graph data, most pairs (993 out of a randomly sampled 1000 phrase pairs) have not been observed in our data (they fell below our thresholds to be considered either a positive or a negative pair in DENPHRASE). This data allows us to test DENENCODE's probability predictions on infrequent events. (We look only at the predicted probabilities $\hat{P}^1_{[\![]\!]}$ because most of the gold denotational probabilities $P^1_{[\![]\!]}$ between these phrase pairs would be 0. We also omit $\hat{P}^2_{[\![]\!]}$ similarities.)

The resulting nearest neighbors, examples of which are shown in Table 7.9, are much noisier for $\hat{P}^1_{[\![]\!]}$ than for GLOVE or SNLI. For examples, the fifth nearest neighbor of *child play stringed instrument* is *child play in sprinkler*. While both phrases contain *child play*, they involve completely different scenes. GLOVE and SNLI, however, do not appear to be so noisy. They appear to rely more on lexical relationships than the denotational phrase model does, so the relationship between the target phrase and the neighbor phrase is usually obvious (*child play stringed instrument* → *girl play stringed instrument*, *woman leap* → *female jump*, *two person discuss* → *two person write*). $\hat{P}^1_{[\![]\!]}$, on the other hand, lacks negative labels and also tends to predict overly high probabilities for long, specific phrases, even when the neighbor and target phrases are clearly unrelated (*two person discuss* → *two person play soccer on field*).

Despite the noise present in $\hat{P}^1_{[\![]\!]}$, there are interesting patterns that continue to indicate that denotational similarity represents different semantic relationships from those modeled by distributional representations. For the target phrase *child play stringed instrument*, $\hat{P}^1_{[\![]\!]}$ includes phrases about *guitars* in its nearest neighbors, while the non-denotational neighbor phrases all involve the phrase *stringed instrument*. $\hat{P}^1_{[\![]\!]}$ is still able to identify hypernyms as being highly relevant even when considering unobserved event pairs. In the second example, *woman leap*, $\hat{P}^1_{[\![]\!]}$ identifies some similar phrases that the non-denotational models miss (*athlete leap*, *woman jump into air*). In the third example, *woman put on makeup*, $\hat{P}^1_{[\![]\!]}$ differs from the non-denotational models GLOVE and SNLI in its focus on other descriptions of women's appearances that probably occur in the same scenes as the target phrase (*woman with blond hair and clothing*, *woman with long black hair*, *woman with ponytail*). Finally, $\hat{P}^1_{[\![]\!]}$'s neighbors for *two person discuss* contain phrases describing what else people could be doing while discussing (*two person sit next to each other*, *two person look at each other*, *two woman and man stand*), in contrast to the synonyms or the subject-repeating phrases that populate the non-denotational neighbors.

	Model	Nearest Neighbors
child play stringed instrument	$\hat{P}^1_{[\![]\!]}$	child play guitar, child play instrument, two man play stringed instrument on stage, boy play guitar, child play in sprinkler, boy play in sand, child play with block, child play on playground
	GloVe	person play stringed instrument, girl play stringed instrument, one play stringed instrument, child play instrument, play stringed instrument, play stringed instrument with person, three person play stringed instrument, man play stringed instrument
	SNLI	child play instrument, girl play stringed instrument, person play stringed instrument, one play stringed instrument, play stringed instrument, clothing play stringed instrument, young woman play stringed instrument, musician play stringed instrument
woman leap	$\hat{P}^1_{[\![]\!]}$	girl leap, athlete leap, newlywed hold bouquet, woman put on makeup, two woman walk down city street, woman jump into air, hiker with backpack, black animal leap, cyclist ride through forest, woman stand in front of car
	GloVe	woman jump, girl leap, young woman jump, woman, embrace woman, woman turn, woman climb, woman point, woman and woman sit, woman reach
	SNLI	woman jump, girl leap, woman climb, young woman jump, female jump, woman reach, woman climb up, woman crouch, woman turn, woman hit
woman put on makeup	$\hat{P}^1_{[\![]\!]}$	on makeup, person apply makeup, clown makeup, woman with blond hair and clothing, woman with long black hair, newlywed hold bouquet, lady with brown hair, woman with long blond hair, woman with ponytail, person with long hair play stringed instrument
	GloVe	woman on right, on woman, on makeup, some woman look, woman sit on, woman look down, woman with clothing look, woman look at her, woman with shirt on, woman with hair
	SNLI	on makeup, person apply makeup, woman with clothing look, woman on right, woman with hair, woman work on computer, woman with shirt on, woman with haircut, woman have tattoo, woman look at her
two person discuss	$\hat{P}^1_{[\![]\!]}$	two person on oppose team, two man discuss, two man play stringed instrument on stage, two person sit next to each other, two woman have conversation, two person play soccer on field, two woman and man stand, two man have discussion, two person look at each other, two guy talk
	GloVe	two person examine, two person have discussion, three person examine, two person have conversation, two other person, two person make, three person take, three other person, two person help, two person come
	SNLI	group of person discuss, two person examine, two person have discussion, three person examine, two person have conversation, two man discuss, person and person talk, two person chat, two person write, three person participate

Table 7.9: Nearest neighbors among 10,000 randomly selected phrases.

7.4 CONCLUSION

In this chapter, we presented a new denotational embedding model that does not require constituent-identification preprocessing and still extends to long sentences, outperforming standard neural sequence models on textual entailment in MPE. This model demonstrates again that denotational probabilities provide beneficial information for textual entailment models, this time on our multiple premise entailment task.

Although our denotational model does not perform quite as well as a similar model that is pretrained on SNLI, the two models make very different types of errors. Additional analysis of the denotational embedding model shows that it successfully learns semantic relationships that are distinct from distributional representations. This difference, which is due to the denotational knowledge of common images that we used to seed the embedding representation, may explain why the denotational model outperforms even the SNLI-pretrained model in disambiguated NEUTRAL items in the context of an unseen image.

CHAPTER 8: CONCLUSION

This thesis focuses on modeling textual entailment in the context of image-grounded descriptions. We started with the concept of denotation, a novel similarity metric, from our earlier work in Young et al. [9], and introduced new models that produce denotational embeddings that can be applied to new tasks. We also presented a new textual entailment corpus that contains sentences linked by a common image denotation, challenging models to build summary scene representations. The experiments we have presented demonstrate that denotational similarity captures different semantic relationships from traditional distributional representations, and that this information can be useful to various semantic tasks.

In Chapter 4, we applied denotational similarities to textual entailment for the first time. These denotational similarities are computed over the constituent phrases in each sentence that correspond to phrases in the denotation graph. Our handcrafted feature model that included denotational features outperformed all other models on textual entailment in the shared task competition. We demonstrate that denotational similarities are complementary to count-based distributional similarities.

In Chapter 5, we introduced a denotational embedding model that expresses denotational set relationships. This model produces denotationally informed phrase embeddings for phrases unseen during training, allowing us to measure denotational similarity between phrases not present in the denotation graph. Furthermore, we demonstrated that these resulting embeddings can be informative for textual entailment.

In Chapter 6, we introduced a new textual entailment task based on the multiple image descriptions available in Flickr30K. This task presents a challenge for models to aggregate information from multiple partially overlapping premise sentences, building an entire coherent scene representation in order to make the correct inference. We use this task to evaluate the effectiveness of standard sentence encoding models as well as the denotational embedding model that we introduce in Chapter 7.

In Chapter 7, we also present an analysis of the phrase embeddings produced by the denotational embedding model. We examine how predicted denotational similarities differ from similarity according to a distributional representation. This investigation shows that the denotational model successfully reproduces the relations in the denotation graph for events with sufficient frequency in Flickr30K. However, the model fails to generalize to infrequent events. Nevertheless, the denotational similarity metric emphasizes different semantic relationships from a distributional similarity metric, relationships that could be useful to other types of downstream tasks, not just textual entailment. It captures scene-grounded infor-

mation rather than synonyms, and contains visual scene information that is not available to text-based models.

8.1 AVENUES FOR FUTURE WORK

Although we have demonstrated the usefulness of denotational similarities on multiple textual entailment datasets, they do have limitations. In this section, we discuss several of these areas for future work, illustrated with examples from MPE.

8.1.1 Domain Adaptation

In the denotation graph, the phrases with the largest denotations are shorter and more general phrases. As a result, denotational similarities are often reliable only for short phrases with a relatively limited vocabulary. In this thesis, we ameliorated this weakness somewhat by using neural phrase encoding models we initialized with pre-trained word embeddings. However, the fundamental limitation remains, and hinders the general applicability of this approach to entailment in other domains. Future work could investigate how to improve the coverage of the current denotation graph and denotational models. That could involve modeling paraphrases in the current image-caption graph in order to improve its density. Another possible approach would be to explore whether it is feasible to automatically build similar denotation graphs in other domains in order to extend this similarity metric to other problems.

8.1.2 Phrase Decomposition

Since denotational information exists primarily for shorter phrases, we found it helpful to first decompose sentences into shorter phrases. In Chapter 4, this involved a parsing pre-processing step to identify constituent phrases. In Chapter 7, we explored multiple approaches to decomposing a sentence into phrases: sliding windows, subsequence phrases, and phrases from the denotation graph. However, all of these methods are preprocessing steps that separate the decomposition step from the entailment prediction step. A model that instead learns its own decomposition method in conjunction with the entailment classification task could identify only the most informative phrases for entailment between any premise and hypothesis sentences.

One possible approach could be to learn an alignment model specifically for predicting entailment in MPE. Existing alignment models have been trained primarily for machine

translation and do not produce useful results on MPE, but it should be useful to have a learned alignment between premises or between the premises and hypothesis in order to identify key phrases for consideration.

8.1.3 Common Sense Knowledge

One weakness of our approach to denotation is that it is difficult to distinguish between a pair of events that *rarely co-occurs but could conceivably do so* and a pair of events that *could not co-occur*. According to how we compute denotational similarity, both event pairs will be classified as unobserved, but we do not have enough information to say *why*. As a result, it is difficult to distinguish between the following two examples using denotational information alone. It is possible to *laugh* and *cook* at the same time even if there are no examples of such a scene in FLICKR30K. On the other hand, people who are *standing* cannot also be *sitting*: this is physically impossible. However, it is difficult to use corpus statistics to definitively separate *mutually exclusive* events from events that are simply unlikely to co-occur.

PREMISES:
A man in black rimmed glasses and a blue button down shirt and a woman in a black and white tank top laugh together.
A man and a woman are laughing.
Two friends having a laugh.
A man and woman laughing.

HYPOTHESIS: A man cooking. ⇒NEUTRAL

PREMISES:
Six casually dressed people watch from the comfort and protection of a wooden rail while a restaurant employee stands in the background.
A group of six people are standing next to a white fence.
Six people stand at the railing at the El Tambor.
Crowd of people standing near a bench.

HYPOTHESIS: A group sits. ⇒CONTRADICTION

We can think of this issue – distinguishing a true negative event pair from an event pair that we did not observe in the data – as the problem of modeling the *unknown unknowns*. Solving this issue requires additional information that could explain what makes two events mutually exclusive.

In constructing the denotation graph, we have assumed that we can learn common sense, real world relationships from linguistic expressions that are grounded in common scenarios, but not otherwise labeled with more fine-grained co-occurrence or causal information. In contrast, there is a line of work regarding common sense knowledge that explicitly labels these unstated assumptions. These works use a purely linguistic approach to represent likely intents and reactions [95], mental states [96], and subjective roles [97] of actors in a particular scene. In addition, Zellers et al. [98] present a dataset that uses adversarial filtering to remove co-occurrence relationships that are easily predicted by existing models, therefore selecting for typically unstated common sense assumptions that are difficult for our models to capture.

These common sense datasets seek to present inference problems that humans can solve easily given their implicit knowledge of the world. Their purely linguistic, explicit labeling approach may be complementary to our grounded, unlabeled approach. Future work could investigate whether different types of common sense information regarding these often unstated *unknown unknowns* can be shared across these datasets and models.

8.1.4 Multiple Premise Handling

Finally, we did not fully address one of the most interesting parts of MPE: combining information from different premises. We tried several approaches, applying attention to different combinations of premise sentences as well as to constituent phrases across premises. However, we did not see any gains from these different approaches.

Ideally, a model would be able to build a unified scene representation from an arbitrary number of sentences, which it could then compare to the hypothesis to judge its compatibility with the entire scene. However, even if we set aside the question of scene representation and focus on entailment prediction, there are still multiple ways that premises may interact with one another and with the hypothesis. In the rest of this section, we present some examples and discuss the difficulties of developing a general solution.

Single Premise

In this case, only one of the four premises contains the information to produce the correct entailment label. This means that a majority voting strategy will not work, as the other three premises have a different relationship with the hypothesis (e.g. three premises are neutral to the hypothesis but the fourth premise entails the hypothesis).

In the following example, only the third premise indicates that the dogs are interacting

with one another in a way that should be interpreted as *greeting each other*. The remaining three premises mention the dogs but do not explain what they are doing.

PREMISES:
Two little dogs are lying on the green grass and a larger dog looks down on them.
A large orange dog and a little orange dog with another third dog in the grass.
Three dogs sniffing each other.
Two dogs and a puppy.

HYPOTHESIS: Dogs greeting each other. ⇒ENTAILMENT

Multiple Premise Reinforcement

In this case, more than one premise contains sufficient information to produce the correct entailment label. It is possible, but not guaranteed, that a majority voting approach will work here (there could still be a 2-2 tie over pairwise entailment relationships). Here, the presence of multiple premises with the same, correct entailment relationship to the hypothesis should serve as a reinforcing signal to produce that entailment label.

In the following example, the first and fourth premises state that the group is performing or singing, which contradicts the idea that the group is having a meeting. These two premises both contradict the hypothesis, and so they reinforce the conclusion that the correct label is CONTRADICTION. The remaining two premises are neutral to the hypothesis (the hypothesis could conceivably be true given the second or third premises), but they are overruled by the two premises that contradict the hypothesis.

PREMISES:
A group of people are standing at the front of the room, preparing to sing.
A group of women with black binders stand in front of a group of people.
Woman standing in front of group with black folders in hand.
A group of individuals performed in front of a seated crowd.

HYPOTHESIS: A group having a meeting. ⇒CONTRADICTION

Multiple Premise Aggregation

In this case, no single premise alone contains sufficient information to predict the overall relationship with the hypothesis. Information must be aggregated from at least two premise sentences in order to predict the correct label.

In the following example, all the premises imply that the person is probably standing up. However, only the fourth premise describes the person as a woman, and only the first and third premises describe a wooded or forested area. We must combine information from the fourth premise and the first or third premise in order to determine that the hypothesis is entailed by the premises.

PREMISES:
A person in a green jacket and pants appears to be digging in a wooded field with several cars in the background.
A young child in a green jacket rakes leaves.
A young child rakes leaves in a wooded area.
A woman cleaning up a park.

HYPOTHESIS: A woman standing in the forest. ⇒ENTAILMENT

We tried several models that reweighted various combinations of premise sentence representations in order to combine information across sentences. In Section 7.1.3, we described several model variants that a) max pool over all four premise vectors, b) apply attention weights over all four premise vectors, or c) max pool over combinations of one to four premise vectors and apply attention weights to the resulting summary vectors. Ultimately, these models, which explicitly combine summary premise vectors, did not outperform a model that uses each premise vector separately to make an entailment label prediction. There is still room for future development here.

The work in this thesis has demonstrated the promise of grounded image-based representations of linguistic expressions and their applicability to important tasks like textual entailment. Much work remains to be done to make denotational similarity useful to a broader spectrum of tasks in multiple domains. However, promising work in areas like paraphrasing, multimodal representations, and common sense modeling could all potentially facilitate further development.

REFERENCES

[1] I. Dagan and O. Glickman, "Probabilistic textual entailment: Generic applied modeling of language variability," 2004.

[2] I. Dagan, O. Glickman, and B. Magnini, "The PASCAL Recognising Textual Entailment challenge," in *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, J. Quiñonero-Candela, I. Dagan, B. Magnini, and F. d'Alché Buc, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 177–190.

[3] M. Marelli, S. Menini, M. Baroni, L. Bentivogli, R. Bernardi, and R. Zamparelli, "A SICK cure for the evaluation of compositional distributional semantic models," in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, 2014.

[4] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 632–642.

[5] I. Dagan, D. Roth, M. Sammons, and F. M. Zanzotto, "Recognizing textual entailment: Models and applications," *Synthesis Lectures on Human Language Technologies*, vol. 6, no. 4, pp. 1–220, 2013.

[6] S. R. Bowman, "Modeling natural language semantics in learned representations," Ph.D. dissertation, Stanford University, 2016.

[7] D. Giampiccolo, H. T. Dang, B. Magnini, I. Dagan, E. Cabrio, and W. B. Dolan, "The Fourth PASCAL Recognizing Textual Entailment Challenge," in *Proceedings of the TAC 2008 Workshop on Textual Entailment*, 2008.

[8] M. Marelli, L. Bentivogli, M. Baroni, R. Bernardi, S. Menini, and R. Zamparelli, "SemEval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment," in *Proceedings of SemEval 2014: International Workshop on Semantic Evaluation*, 2014.

[9] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier, "From image descriptions to visual denotations," *Transactions of the Association of Computational Linguistics – Volume 2, Issue 1*, pp. 67–78, 2014.

[10] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. Bernstein, and L. Fei-Fei, "Visual Genome: Connecting language and vision using crowdsourced dense image annotations," 2016.

[11] S. Gururangan, S. Swayamdipta, O. Levy, R. Schwartz, S. Bowman, and N. A. Smith, "Annotation artifacts in natural language inference data," in *Proceedings of the 2018*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers). New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 107–112.

[12] A. Poliak, J. Naradowsky, A. Haldar, R. Rudinger, and B. Van Durme, "Hypothesis only baselines in natural language inference," in *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 180–191.

[13] R. Adams, G. Nicolae, C. Nicolae, and S. Harabagiu, "Textual entailment through extended lexical overlap and lexico-semantic matching," in *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Association for Computational Linguistics, 2007, pp. 119–124.

[14] M.-C. de Marneffe, T. Grenager, B. MacCartney, D. Cer, D. Ramage, C. Kiddon, and C. D. Manning, "Aligning semantic graphs for textual inference and machine reading," in *AAAI Spring Symposium at Stanford*, 2007.

[15] M. Heilman and N. A. Smith, "Tree edit models for recognizing textual entailments, paraphrases, and answers to questions," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Los Angeles, California: Association for Computational Linguistics, June 2010, pp. 1011–1019.

[16] J. R. Hobbs, M. Stickel, P. Martin, and D. Edwards, "Interpretation as abduction," in *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics*. Buffalo, New York, USA: Association for Computational Linguistics, June 1988, pp. 95–103.

[17] E. Akhmatova, "Textual entailment resolution via atomic propositions," in *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment*. Citeseer, 2005.

[18] J. Bos and K. Markert, "When logical inference helps determining textual entailment (and when it doesn't)," in *Proceedings of the Second PASCAL RTE Challenge*, 2006, p. 26.

[19] R. Raina, A. Y. Ng, and C. D. Manning, "Robust textual inference via learning and abductive reasoning," in *AAAI*, 2005, pp. 1099–1105.

[20] I. Beltagy, S. Roller, G. Boleda, K. Erk, and R. Mooney, "UTexas: Natural language semantics using distributional semantics and probabilistic logic," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, August 2014, pp. 796–801.

[21] Y. Bestgen, "CECL: a new baseline and a non-compositional approach for the SICK benchmark," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Association for Computational Linguistics, 2014, pp. 160–165.

[22] E. Bicici and A. Way, "RTM-DCU: Referential translation machines for semantic similarity," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, August 2014, pp. 487–496.

[23] J. Bjerva, J. Bos, R. van der Goot, and M. Nissim, "The Meaning Factory: Formal semantics for recognizing textual entailment and determining semantic similarity," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, August 2014, pp. 642–646.

[24] L. Ferrone and F. M. Zanzotto, "haLF: Comparing a pure CDSM approach with a standard machine learning system for RTE," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, August 2014, pp. 300–304.

[25] R. Gupta, H. Bechara, I. El Maarouf, and C. Orasan, "UoW: NLP techniques developed at the University of Wolverhampton for semantic similarity and textual entailment," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Association for Computational Linguistics, 2014, pp. 785–789.

[26] S. Jimenez, G. Dueñas, J. Baquero, and A. Gelbukh, "UNAL-NLP: Combining soft cardinality features for semantic textual similarity, relatedness and entailment," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, August 2014, pp. 732–742.

[27] S. Leon, D. Vilariño, D. Pinto, M. Tovar, and B. Beltrán, "BUAP: Evaluating compositional distributional semantic models on full sentences through semantic relatedness and textual entailment," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Association for Computational Linguistics, 2014, pp. 145–148.

[28] E. Lien and M. Kouylekov, "UIO-Lien: Entailment recognition using minimal recursion semantics," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Association for Computational Linguistics, 2014, pp. 699–703.

[29] T. Proisl, S. Evert, P. Greiner, and B. Kabashi, "SemantiKLUE: Robust semantic similarity at multiple levels using maximum weight matching," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, August 2014, pp. 532–540.

[30] N. P. A. Vo, O. Popescu, and T. Caselli, "FBK-TR: SVM for semantic relatedeness and corpus patterns for RTE," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, August 2014, pp. 289–293.

[31] J. Zhao, T. Zhu, and M. Lan, "ECNU: One stone two birds: Ensemble of heterogenous measures for semantic relatedness and textual entailment," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, August 2014, pp. 271–277.

[32] I. Beltagy, S. Roller, P. Cheng, K. Erk, and R. J. Mooney, "Representing Meaning with a Combination of Logical and Distributional Models," *Computational Linguistics*, vol. 42, no. 4, pp. 763–808, Dec. 2016.

[33] W. Yin, H. Schütze, B. Xiang, and B. Zhou, "ABCNN: Attention-based convolutional neural network for modeling sentence pairs," *Transactions of the Association of Computational Linguistics*, vol. 4, pp. 259–272, 2016.

[34] W. Yin and H. Schütze, "Task-specific attentive pooling of phrase alignments contributes to sentence matching," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, April 2017, pp. 699–709.

[35] A. Lai and J. Hockenmaier, "Illinois-LH: A denotational and distributional approach to semantics," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 2014, pp. 329–334.

[36] J. L. Elman, "Distributed representations, simple recurrent networks, and grammatical structure," *Machine Learning*, vol. 7, no. 2, pp. 195–225, Sep 1991.

[37] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Compututation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[38] L. Sha, B. Chang, Z. Sui, and S. Li, "Reading and thinking: Re-read LSTM unit for textual entailment recognition," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan: The COLING 2016 Organizing Committee, December 2016, pp. 2870–2879.

[39] T. Munkhdalai and H. Yu, "Neural semantic encoders," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, April 2017, pp. 397–407.

[40] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Copenhagen, Denmark: Association for Computational Linguistics, September 2017, pp. 670–680.

[41] Q. Chen, X. Zhu, Z.-H. Ling, S. Wei, H. Jiang, and D. Inkpen, "Enhanced LSTM for natural language inference," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017, pp. 1657–1668.

[42] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 2227–2237.

[43] S. R. Bowman, J. Gauthier, A. Rastogi, R. Gupta, C. D. Manning, and C. Potts, "A fast unified model for parsing and sentence understanding," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, August 2016, pp. 1466–1477.

[44] L. Mou, R. Men, G. Li, Y. Xu, L. Zhang, R. Yan, and Z. Jin, "Natural language inference by tree-based convolution and heuristic matching," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Berlin, Germany: Association for Computational Linguistics, August 2016, pp. 130–136.

[45] T. Munkhdalai and H. Yu, "Neural tree indexers for text understanding," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, April 2017, pp. 11–21.

[46] T. Rocktäschel, E. Grefenstette, K. M. Hermann, T. Kocisky, and P. Blunsom, "Reasoning about entailment with neural attention," in *The International Conference on Learning Representations (ICLR)*, 2016.

[47] A. Parikh, O. Täckström, D. Das, and J. Uszkoreit, "A decomposable attention model for natural language inference," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, November 2016, pp. 2249–2255.

[48] A. S. White, P. Rastogi, K. Duh, and B. Van Durme, "Inference is everything: Recasting semantic resources into a unified evaluation framework," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, November 2017, pp. 996–1005.

[49] M. Hodosh, P. Young, and J. Hockenmaier, "Framing image description as a ranking task: Data, models and evaluation metrics," *Journal of Artificial Intelligence Research*, vol. 47, no. 1, pp. 853–899, May 2013.

[50] R. Montague, *Formal Philosophy; Selected Papers of Richard Montague*. New Haven: Yale University Press, 1974.

[51] D. Dowty, R. Wall, and S. Peters, *Introduction to Montague Semantics*. Dordrecht: Reidel, 1981.

[52] J. Barwise and J. Perry, "Situations and attitudes," *Journal of Philosophy*, vol. 78, no. 11, pp. 668–691, 1981.

[53] P. Young, "Using the visual denotations of image captions for semantic inference," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2013.

[54] J. Nivre, J. Hall, and J. Nilsson, "Maltparser: A data-driven parser-generator for dependency parsing," in *Proceedings of LREC*, vol. 6, 2006, pp. 2216–2219.

[55] G. A. Miller, "WordNet: A Lexical Database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995.

[56] M. Hodosh, P. Young, C. Rashtchian, and J. Hockenmaier, "Cross-caption coreference resolution for automatic image understanding," in *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*. Uppsala, Sweden: Association for Computational Linguistics, July 2010, pp. 162–171.

[57] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of english: The penn treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, Jun. 1993.

[58] R. Parker, D. Graff, J. Kong, K. Chen, and K. Maeda, "English Gigaword Fifth Edition." Linguistic Data Consortium, 2011.

[59] A. K. McCallum, "Mallet: A machine learning for language toolkit," 2002. [Online]. Available: http://mallet.cs.umass.edu

[60] E. Agirre, M. Diab, D. Cer, and A. Gonzalez-Agirre, "SemEval-2012 task 6: A pilot on semantic textual similarity," in *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 2012, pp. 385–393.

[61] D. Chen and W. Dolan, "Collecting highly parallel data for paraphrase evaluation," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 190–200.

[62] D. Bär, T. Zesch, and I. Gurevych, "DKPro Similarity: An open source framework for text similarity," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Sofia, Bulgaria: Association for Computational Linguistics, August 2013, pp. 121–126.

[63] E. Gabrilovich and S. Markovitch, "Computing semantic relatedness using Wikipedia-based Explicit Semantic Analysis," in *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, ser. IJCAI'07. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007, pp. 1606–1611.

[64] P. Resnik, "Using information content to evaluate semantic similarity in a taxonomy," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, ser. IJCAI'95. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, pp. 448–453.

[65] J. Berant, I. Dagan, and J. Goldberger, "Learning entailment relations by global graph structure optimization," *Computational Linguistics*, vol. 38, no. 1, pp. 73–111, Mar. 2012.

[66] L. Kotlerman, I. Dagan, B. Magnini, and L. Bentivogli, "Textual Entailment Graphs," *Natural Language Engineering*, pp. 1–25, Jan 2015.

[67] J. Mitchell and M. Lapata, "Composition in distributional models of semantics," *Cognitive Science*, vol. 34, no. 8, pp. 1388–1429, 2010.

[68] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, 1970.

[69] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[70] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *SIGKDD Explorations*, vol. 11, no. 1, pp. 10–18, 2009.

[71] V. Punyakanok and D. Roth, "The use of classifiers in sequential inference." in *NIPS*. MIT Press, 2001, pp. 995–1001.

[72] R. Cooper, R. Crouch, J. van Eijck, C. Fox, J. van Genabith, J. Jaspars, H. Kamp, M. Pinkal, D. Milward, M. Poesio, S. Pulman, T. Briscoe, H. Maier, and K. Konrad, "Using the framework," FraCaS: A Framework for Computational Semantics, Tech. Rep., 1996.

[73] A. Lai and J. Hockenmaier, "Learning to predict denotational probabilities for modeling entailment," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, April 2017, pp. 721–730.

[74] G. Kruszewski, D. Paperno, and M. Baroni, "Deriving boolean structures from distributional vectors," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 375–388, 2015.

[75] J. Henderson and D. Popa, "A vector space for distributional semantics for entailment," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany, August 2016, pp. 2052–2062.

[76] K. Erk, "Representing words as regions in vector space," in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, Boulder, Colorado, June 2009, pp. 57–65.

[77] L. Vilnis and A. McCallum, "Word representations via gaussian embedding," in *The International Conference on Learning Representations (ICLR)*, 2015.

[78] I. Vendrov, R. Kiros, S. Fidler, and R. Urtasun, "Order-embeddings of images and language," in *The International Conference on Learning Representations (ICLR)*, 2016.

[79] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.

[80] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2014.

[81] L. Vilnis, X. Li, S. Murty, and A. McCallum, "Probabilistic embedding of knowledge graphs with box lattice measures," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2018, pp. 263–272.

[82] A. Lai, Y. Bisk, and J. Hockenmaier, "Natural language inference from multiple premises," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, November 2017, pp. 100–109.

[83] Y. Nie and M. Bansal, "Shortcut-stacked sentence encoders for multi-domain inference," in *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP*. Copenhagen, Denmark: Association for Computational Linguistics, September 2017, pp. 41–45.

[84] M. Baroni, G. Dinu, and G. Kruszewski, "Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland: Association for Computational Linguistics, June 2014, pp. 238–247.

[85] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[86] O. Levy and Y. Goldberg, "Linguistic regularities in sparse and explicit word representations," in *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*. Ann Arbor, Michigan: Association for Computational Linguistics, June 2014, pp. 171–180.

[87] O. Levy, Y. Goldberg, and I. Dagan, "Improving distributional similarity with lessons learned from word embeddings," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 211–225, 2015.

[88] H. Liu, F. Hussain, C. L. Tan, and M. Dash, "Discretization: An enabling technique," *Data Mining and Knowledge Discovery*, vol. 6, no. 4, pp. 393–423, Oct 2002.

[89] G. Charpiat, M. Hofmann, and B. Schölkopf, "Automatic image colorization via multimodal predictions," in *Computer Vision – ECCV 2008*, D. Forsyth, P. Torr, and A. Zisserman, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 126–139.

[90] S. Garca, J. Luengo, J. A. Sez, V. Lpez, and F. Herrera, "A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 4, pp. 734–750, April 2013.

[91] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 577–593.

[92] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep Ordinal Regression Network for Monocular Depth Estimation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[93] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in pretraining distributed word representations," in *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[94] H. Vu, C. Greco, A. Erofeeva, S. Jafaritazehjan, G. Linders, M. Tanti, A. Testoni, R. Bernardi, and A. Gatt, "Grounded textual entailment," in *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, 2018, pp. 2354–2368.

[95] H. Rashkin, M. Sap, E. Allaway, N. A. Smith, and Y. Choi, "Event2Mind: Commonsense inference on events, intents, and reactions," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 463–473.

[96] H. Rashkin, A. Bosselut, M. Sap, K. Knight, and Y. Choi, "Modeling naive psychology of characters in simple commonsense stories," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2018, pp. 2289–2299.

[97] H. Rashkin, S. Singh, and Y. Choi, "Connotation frames: A data-driven investigation," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2016, pp. 311–321.

[98] R. Zellers, Y. Bisk, R. Schwartz, and Y. Choi, "SWAG: A large-scale adversarial dataset for grounded commonsense inference," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, October-November 2018, pp. 93–104.