Learning to Predict Denotational Probabilities For Modeling Entailment

Alice Lai and Julia Hockenmaier
Department of Computer Science
University of Illinois at Urbana-Champaign
{aylai2, juliahmr}@illinois.edu

Abstract

We propose a framework that captures the denotational probabilities of words and phrases by embedding them in a vector space, and present a method to induce such an embedding from a dataset of denotational probabilities. We show that our model successfully predicts denotational probabilities for unseen phrases, and that its predictions are useful for textual entailment datasets such as SICK and SNLI.

1 Introduction

In order to bridge the gap between vector-based distributional approaches to lexical semantics that are intended to capture which words occur in similar contexts, and logic-based approaches to compositional semantics that are intended to capture the truth conditions under which statements hold, Young et al. (2014) introduced the concept of "denotational similarity." Denotational similarity is intended to measure the similarity of simple, declarative statements in terms of the similarity of their truth conditions.

From classical truth-conditional semantics, Young et al. borrowed the notion of the denotation of a declarative sentence s, $[\![s]\!]$, as the set of possible worlds in which the sentence is true. Young et al. apply this concept to the domain of image descriptions by defining the *visual denotation* of a sentence s as the set of images that s describes. The denotational probability of s, $P_{[\![]\!]}(s)$, is the number of images in the visual denotation of s over the size of the corpus. Two sentences are denotationally similar if the sets of images (possible worlds) they describe have a large overlap. For example, "A woman is jogging on a beach" and "A woman is running on a sandy shore" can often be used to describe the same scenario, so they will have a large image overlap that corresponds to high denotational similarity.

Given the above definitions, Young et al. estimate the denotational probabilities of phrases from FLICKR30K, a corpus of 30,000 images, each paired with five descriptive captions. Young et al. (2014) and Lai and Hockenmaier (2014) showed that these similarities are complementary to standard distributional similarities, and potentially more useful for semantic tasks that involve entailment. However, the systems presented in these papers were restricted to looking up the denotational similarities of frequent phrases in the training data. In this paper, we go beyond this prior work and define a model that can *predict* the denotational probabilities of novel phrases and sentences. Our experimental results indicate that these predicted denotational probabilities are useful for several textual entailment datasets.

2 Textual entailment in SICK and SNLI

The goal of textual entailment is to predict whether a hypothesis sentence is true, false, or neither based on the premise text (Dagan et al., 2013). Due in part to the Recognizing Textual Entailment (RTE) challenges (Dagan et al., 2006), the task of textual entailment recognition has received a lot of attention in recent years. Although full entailment recognition systems typically require a complete NLP pipeline, including coreference resolution, etc., this paper considers a simplified variant of this task in which the premise and hypothesis are each a single sentence. This simplified task allows us to ignore the complexities that arise in longer texts, and instead focus on the purely semantic problem of how to represent the meaning of sentences. This version of the textual entailment task has been popularized by two datasets, the Sentences Involving Compositional Knowledge (SICK) dataset (Marelli et al., 2014) and the Stanford Natural Language Inference (SNLI) corpus (Bowman et al., 2015), both of which involve a 3-way classification for textual entailment.

SICK was created for SemEval 2014 based on image caption data and video descriptions. The premises and hypotheses are automatically generated from the original captions and so contain some unintentional systematic patterns. Most approaches to SICK involve hand-engineered features (Lai and Hockenmaier, 2014) or large collections of entailment rules (Beltagy et al., 2015).

SNLI is the largest textual entailment dataset by several orders of magnitude. It was created with the goal of training neural network models for textual entailment. The premises in SNLI are captions from the FLICKR30K corpus (Young et al., 2014). The hypotheses (entailed, contradictory, or neutral in relation to the premise) were solicited from workers on Mechanical Turk. Bowman et al. (2015) initially illustrated the effectiveness of LSTMs (Hochreiter and Schmidhuber, 1997) on SNLI, and recent approaches have focused on improvements in neural network architectures. These include sentence embedding models (Liu et al., 2016; Munkhdalai and Yu, 2017a), neural attention models (Rocktäschel et al., 2016; Parikh et al., 2016), and neural tree-based models (Munkhdalai and Yu, 2017b; Chen et al., 2016). In contrast, in this paper we focus on using a different input representation, and demonstrate its effectiveness when added to a standard neural network model for textual entailment. We demonstrate that the results of the LSTM model of Bowman et al. (2015) can be improved by adding a single feature based on our predicted denotational probabilities. We expect to see similar improvements when our predicted probabilities are added to more complex neural network entailment models, but we leave those experiments for future work.

3 Vector space representations

Several related works have explored different approaches to learning vector space representations that express entailment more directly. Kruszewski et al. (2015) 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 (2016) 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 (2009) 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 (2015) use Gaussian functions to map a word to a density over a latent space. Both papers evaluate their models only on lexical relationships.

4 Denotational similarities

In contrast to traditional distributional similarities, Young et al. (2014) introduced the concept of "denotational similarities" to capture which expressions can be used to describe similar situations. Young et al. first define the *visual denotation* of a sentence (or phrase) s, $[\![s]\!]$, as the (sub)set of images that s can describe. They estimate the denotation of a phrase and the resulting similarities from FLICKR30K, a corpus of 30,000 images, each paired with five descriptive captions. In order to compute visual denotations from the corpus, they define a set of normalization and reduction rules (e.g. lemmatization, dropping modifiers, replacing nouns with their hypernyms, dropping PPs, extracting NPs) that augment the original FLICKR30K captions with a large number of shorter, more generic phrases that are each associated with a subset of the FLICKR30K images.

The result is a large subsumption hierarchy over phrases, which Young et al. call a denotation graph (see Figure 1). The structure of the denotation graph is similar to the idea of an entailment graph (Berant et al., 2012). Each node in the denotation graph corresponds to a phrase s, associated with its denotation $[\![s]\!]$, i.e. the set of images that correspond to the original captions from which this phrase could be derived. For example, the denotation of a phrase *"woman jog on beach"* is the set of images in the corpus that depict a woman jogging on a beach. Note that the deno-

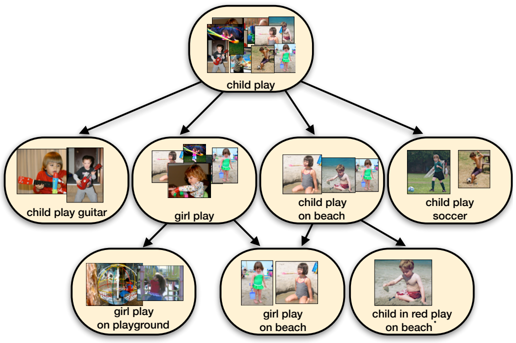


Figure 1: The denotation graph is a subsumption hierarchy over phrases associated with images.

tation of a node (e.g. *"woman jog on beach"*) is always a subset of the denotations of any of its ancestors (e.g. *"woman jog"*, *"person jog"*, *"jog on beach"*, or *"beach"*).

The denotational probability of a phrase s, $P_{\llbracket\rrbracket}(s)$, is a Bernoulli random variable that corresponds to the probability that a randomly drawn image can be described by s. Given a denotation graph over N images, $P_{\llbracket\rrbracket}(s) = \frac{|\llbracket s\rrbracket|}{N}$. The joint denotational probability of two phrases x and y, $P_{\llbracket\rrbracket}(x, y) = \frac{|\llbracket x\rrbracket \cap \llbracket y\rrbracket|}{N}$, indicates how likely it is that a situation can be described by both x and y. Young et al. propose to use pointwise mutual information scores (akin to traditional distributional similarities) and conditional probabilities $P_{\llbracket\rrbracket}(x|y) = \frac{|\llbracket x\rrbracket \cap \llbracket y\rrbracket|}{|\llbracket y\rrbracket|}$ as so-called denotational similarities. In this paper, we will work with denotational conditional probabilities, as they are intended to capture entailment-like relations that hold due to commonsense knowledge, hyponymy, etc. (what is the probability that x is true, given that y can be said about this situation?). In an ideal scenario, if the premise p entails the hypothesis h, then the conditional probability $P(h|p)$ is 1 (or close to 1). Conversely, if h contradicts p, then the conditional probability $P(h|p)$ is close to 0. We therefore stipulate that learning to predict the conditional probability of one phrase h given another phrase p would be helpful in predicting textual entailment. We also note that by the definition of the denotation graph, if x is an ancestor of y in the graph, then y entails x and $P_{\llbracket\rrbracket}(x|y) = 1$.

Young et al. (2014) and Lai and Hockenmaier (2014) show that denotational probabilities can be at least as useful as traditional distributional similarities for tasks that require semantic inference such as entailment or textual similarity recognition. However, their systems can only use denotational probabilities between phrases that already exist in the denotation graph (i.e. phrases that can be derived from the original FLICKR30K captions).

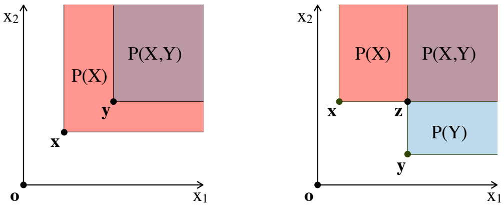


Figure 2: An embedding space that expresses the individual probability of events X and Y and the joint probability $P(X, Y)$.

Here, we present a model that learns to predict denotational probabilities $P_{\llbracket\rrbracket}(x)$ and $P_{\llbracket\rrbracket}(x|y)$ even for phrases it has not seen during training. Our model is inspired by Vendrov et al. (2016), who observed that a partial ordering $\preceq$ over the vector representations of phrases can be used to express an entailment relationship. They induce a so-called order embedding for words and phrases such that the vector $\mathbf{x}$ corresponding to phrase x is smaller than the vector $\mathbf{y}$, i.e. $\mathbf{x} \preceq \mathbf{y}$, for phrases y that are entailed by x, where $\preceq$ corresponds to the reversed product order on R^N_+ ($\mathbf{x} \preceq \mathbf{y} \Leftrightarrow x_i \geq y_i \forall i$). They use their model to predict entailment labels between pairs of sentences, but it is only capable of making a binary entailment decision.

5 An order embedding for probabilities

We generalize this idea to learn an embedding space that expresses not only the binary relation that phrase x is entailed by phrase y, but also the probability that phrase x is true given phrase y. Specifically, we learn a mapping from a phrase x to an N-dimensional vector $\mathbf{x} \in \mathbb{R}^N_+$ such that the vector $\mathbf{x} = (x_1, ..., x_N)$ defines the denotational probability of x as $P_{\llbracket\rrbracket}(x) = \exp(-\sum_i x_i)$. The origin (the zero vector) therefore has probability $\exp(0) = 1$. Any other vector $\mathbf{x}$ that does not lie on the origin (i.e. $\exists_i x_i > 0$) has probability less than 1, and a vector $\mathbf{x}$ that is farther from the origin than a vector $\mathbf{y}$ represents a phrase x that has a smaller denotational probability than phrase y. We can visualize this as each phrase vector occupying a region in the embedding space that is proportional to the denotational probability of the phrase. Figure 2 illustrates this in two dimensions. The zero vector at the origin has a probability proportional to the entire region of the positive orthant, while other points in the space correspond to smaller regions and thus probabilities less than 1.

The joint probability $P_{\llbracket\rrbracket}(x, y)$ in this embedding space should be proportional to the size of the intersection of the regions of $\mathbf{x}$ and $\mathbf{y}$. Therefore, we define the joint probability of two phrases x and y to correspond to the vector $\mathbf{z}$ that is the element-wise maximum of $\mathbf{x}$ and $\mathbf{y}$: $z_i = \max(x_i, y_i)$. This allows us to compute the conditional probability $P_{\llbracket\rrbracket}(x|y)$ as follows:

$$\begin{aligned} P_{\llbracket\rrbracket}(x|y) &= \frac{P_{\llbracket\rrbracket}(x, y)}{P_{\llbracket\rrbracket}(y)} \\ &= \frac{\exp(-\sum_i z_i)}{\exp(-\sum_i y_i)} \\ &= \exp(\sum_i y_i - \sum_i z_i) \end{aligned}$$

Shortcomings We note that this embedding 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 2, 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 $\mathbf{y} = (y_1, ..., y_N)$, $P_{\llbracket\rrbracket}(X, Y) \geq P_{\llbracket\rrbracket}(X)P_{\llbracket\rrbracket}(Y)$:

$$\begin{aligned} P_{\llbracket\rrbracket}(X, Y) &= \exp\big(-\sum_i \max(x_i, y_i)\big) \\ &\geq \exp\big(-\sum_i x_i - \sum_i y_i\big) \\ &= P_{\llbracket\rrbracket}(X)P_{\llbracket\rrbracket}(Y) \end{aligned}$$

(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 our model cannot represent two events with completely disjoint denotations, we will see below that it is able to learn that some phrase pairs have very low denotational conditional probabilities. We note also that our model cannot express $P(X) = 0$ exactly, but can get arbitrarily close in order to represent the probability of a phrase that is extremely unlikely.

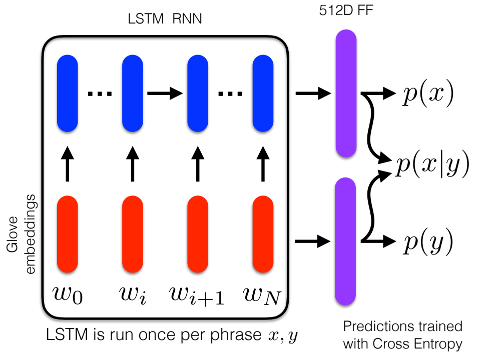


Figure 3: Our probability model architecture. Each phrase is a sequence of word embeddings that is passed through an LSTM to produce a 512d vector representation for the premise and the hypothesis. Both vectors are used to compute the predicted conditional probability and calculate the loss.

6 Our model for $P_{\llbracket\rrbracket}(x)$ and $P_{\llbracket\rrbracket}(x, y)$

We train a neural network model to predict $P_{\llbracket\rrbracket}(x)$, $P_{\llbracket\rrbracket}(y)$, and $P_{\llbracket\rrbracket}(x|y)$ for phrases x and y. This model consists of an LSTM that outputs a 512d vector which is passed through an additional 512d layer. We use 300d GloVe vectors (Pennington et al., 2014) trained on 840B tokens as the word embedding input to the LSTM. We use the same model to represent both x and y regardless of which phrase is the premise or the hypothesis. Thus, we pass the sequence of word embeddings for phrase x through the model to get $\mathbf{x}$, and we do the same for phrase y to get $\mathbf{y}$. As previously described, we sum the elements of $\mathbf{x}$ and $\mathbf{y}$ to get the predicted denotational probabilities $P_{\llbracket\rrbracket}(x)$ and $P_{\llbracket\rrbracket}(y)$. From $\mathbf{x}$ and $\mathbf{y}$, we find the joint vector $\mathbf{z}$, which we use to compute the predicted denotational conditional probability $P_{\llbracket\rrbracket}(x|y)$ according to the equation in Section 5. Figure 3 illustrates the structure of our model.

Our training data consists of ordered phrase pairs $\langle x, y\rangle$. We train our model to predict the denotational probabilities of each phrase ($P_{\llbracket\rrbracket}(x)$ and $P_{\llbracket\rrbracket}(y)$) as well as the conditional probability $P_{\llbracket\rrbracket}(x|y)$. Typically the pair $\langle y, x\rangle$ will also appear in the training data.

Our per-example loss is the sum of the cross entropy losses for $P_{\llbracket\rrbracket}(x)$, $P_{\llbracket\rrbracket}(y)$, and $P_{\llbracket\rrbracket}(x|y)$:

$$\begin{aligned} L = \quad & -\big[P_{\llbracket\rrbracket}(x)\log Q(x) + (1-P_{\llbracket\rrbracket}(x))\log\big(1-Q(x)\big)\big] \\ & -\big[P_{\llbracket\rrbracket}(y)\log Q(y) + (1-P_{\llbracket\rrbracket}(y))\log\big(1-Q(y)\big)\big] \\ & -\big[P_{\llbracket\rrbracket}(x|y)\log Q(x|y) + (1-P_{\llbracket\rrbracket}(x|y))\log\big(1-Q(x|y)\big)\big] \end{aligned}$$

We use the Adam optimizer with a learning rate of 0.001, and a dropout rate of 0.5. These parameters were tuned on the development data.

Numerical issues In Section 5, we described the probability vectors $\mathbf{x}$ as being in the positive orthant. However, in our implementation, we use unnormalized log probabilities. This puts all of our vectors in the negative orthant instead, but it prevents the gradients from becoming too small during training. To ensure that the vectors are in $\mathbb{R}^N_-$, we clip the values of the elements of $\mathbf{x}$ so that $x_i \leq 0$. To compute $\log P_{\llbracket\rrbracket}(x)$, we sum the elements of $\mathbf{x}$ and clip the sum 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 $\log P_{\llbracket\rrbracket}(x|y) = \log P_{\llbracket\rrbracket}(x,y) - \log P_{\llbracket\rrbracket}(y)$, where $\log P_{\llbracket\rrbracket}(x,y)$ is now the element-wise minimum:

$$\log P_{\llbracket\rrbracket}(x,y) = \sum_i \min(x_i, y_i)$$

This element-wise minimum is a standard pooling operation (we take the minimum instead of the more common max pooling). Note that if $x_i > y_i$, neither element x_i nor y_i is updated with respect to the $P_{\llbracket\rrbracket}(x|y)$ loss. Both x_i and y_i will always be updated with respect to the $P_{\llbracket\rrbracket}(x)$ and $P_{\llbracket\rrbracket}(y)$ components of the loss.

6.1 Training regime

To train our model, we use phrase pairs $\langle x, y\rangle$ from the denotation graph generated on the training split of the FLICKR30K corpus (Young et al., 2014). We consider all 271,062 phrases that occur with at least 10 images in the training split of the graph, to ensure that the phrases are frequent enough that their computed denotational probabilities are reliable. Since the FLICKR30K captions are lemmatized in order to construct the denotation graph, all the phrases in the dataset described in this section are lemmatized as well.

We include all phrase pairs where the two phrases have at least one image in common. These constitute 45 million phrase pairs $\langle x, y\rangle$ with $P_{\llbracket\rrbracket}(x|y) > 0$. To train our model to predict $P_{\llbracket\rrbracket}(x|y) = 0$, we include phrase pairs $\langle x, y\rangle$ that have no images in common if $N \times P_{\llbracket\rrbracket}(x)P_{\llbracket\rrbracket}(y) \geq N^{-1}$ (N is the total number of images), meaning that x and y occur frequently enough that we would expect them to co-occur at least once in the data. This yields 2 million pairs where $P_{\llbracket\rrbracket}(x|y) = 0$. For additional examples of $P_{\llbracket\rrbracket}(x|y) = 1$, we include phrase pairs that have an ancestor-descendant relationship in the denotation graph. We include all ancestor-descendant pairs where each phrase occurs with at least 2 images, for an additional 3 million phrase pairs.

For evaluation purposes, we first assign 5% of the phrases to the development pool and 5% to the test pool. The actual test data then consists of all phrase pairs where at least one of the two phrases comes from the test pool. The resulting test data contains 10.6% unseen phrases by type and 51.2% unseen phrases by token. All phrase pairs in the test data contain at least one phrase that was unseen in the training or development data. The development data was created the same way.

This dataset is available to download at `http://nlp.cs.illinois.edu/HockenmaierGroup/data.html`.

We train our model on the training data (42 million phrase pairs) with batch size 512 for 10 epochs, and use the mean KL divergence on the conditional probabilities in the development data to select the best model. Since $P_{\llbracket\rrbracket}(x|y)$ is a Bernoulli distribution, we compute the KL divergence for each phrase pair $\langle x, y\rangle$ as

$$\begin{aligned} D_{KL}(P||Q) = {} & P_{\llbracket\rrbracket}(x|y)\log\frac{P_{\llbracket\rrbracket}(x|y)}{Q(x|y)} \\ & + \big(1 - P_{\llbracket\rrbracket}(x|y)\big)\log\frac{1-P_{\llbracket\rrbracket}(x|y)}{1-Q(x|y)} \end{aligned}$$

where $Q(x|y)$ is the conditional probability predicted by our model.

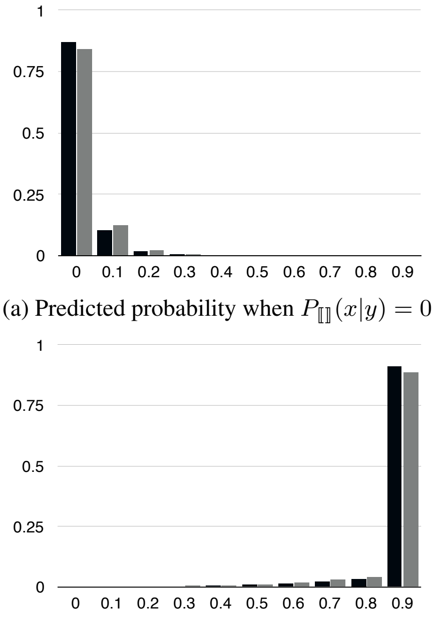
7 Predicting denotational probabilities

7.1 Prediction on new phrase pairs

We evaluate our model using 1) the KL divergences $D_{KL}(P||Q)$ of the gold individual and conditional probabilities $P_{\llbracket\rrbracket}(x)$ and $P_{\llbracket\rrbracket}(x|y)$ against the corresponding predicted probabilities Q, and 2) the Pearson correlation r, which expresses the correlation between two variables (the per-item gold and predicted probabilities) as a value between -1 (total negative correlation) and

	$P(x)$		$P(x\|y)$	
	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 1: Our model predicts the probability of unseen phrase pairs with high correlation to the gold probabilities.



(a) Predicted probability when $P_{\llbracket\rrbracket}(x|y) = 0$

(b) Predicted probability when $P_{\llbracket\rrbracket}(x|y) = 0$

Figure 4: Predicted probabilities on denotational phrase test data when $P_{\llbracket\rrbracket}(x|y) = 0$ is 0 or 1. 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 size of the data.

1 (total positive correlation). As described above, we compute the KL divergence on a per-item basis, and report the mean over all items in the test set.

Table 1 shows the performance of our trained model on unseen test data. The full test data consists of 4.6 million phrase pairs, all of which contain at least one phrase that was not observed in either the training or development data. Our model does reasonably well at predicting these conditional probabilities, reaching a correlation of $r = 0.949$ with $P_{\llbracket\rrbracket}(x|y)$ on the complete test data. On the subset of 123,000 test phrase pairs where both phrases are previously unseen, the model's predictions are almost as good at $r = 0.920$.

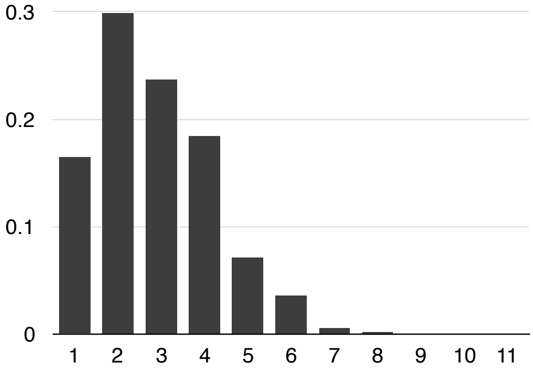


Figure 5: Distribution of phrase lengths as a fraction of the data size on the denotation graph phrase training data.

On the subset of 3,100 test phrase pairs where at least one word was unseen in training, the model's predictions are worse, predicting $P_{\llbracket\rrbracket}(x|y)$ with a correlation of $r = 0.696$. On the remaining test pairs, the model predicts $P_{\llbracket\rrbracket}(x|y)$ with a correlation of $r = 0.949$.

We also analyze our model's accuracy on phrase pairs where the gold $P_{\llbracket\rrbracket}(x|y)$ is either 0 or 1. The latter case reflects an important property of the denotation graph, since $P_{\llbracket\rrbracket}(x|y) = 1$ when x is an ancestor of y. More generally, we can interpret $P_{\llbracket\rrbracket}(h|p) = 1$ as a confident prediction of entailment, and $P_{\llbracket\rrbracket}(h|p) = 0$ as a confident prediction of contradiction. Figure 4 shows the distribution of predicted conditional probabilities for phrase pairs where gold $P_{\llbracket\rrbracket}(h|p) = 0$ (top) and gold $P_{\llbracket\rrbracket}(h|p) = 1$ (bottom). Our model's predictions on unseen phrase pairs (gray bars) are nearly as accurate as its predictions on the full test data (black bars).

7.2 Prediction on longer sentences

Our model up to this point 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 for the distribution of phrase lengths in the training data). To improve our model's performance on longer sentences, we add the SNLI training data (which has a mean sentence length of 11 words) to our training data. We train a new model from scratch on a corpus consisting of the previously described 42 million phrase pairs and the 550,000 SNLI training sentence pairs (lemmatized to match our phrase pairs). We do not train on SICK because the corpus is much smaller and has a different distribution of phenomena, including explicit negation. We augment the SNLI data with approximate gold denotational probabilities by assigning a probability $P_{[\![]\!]}(S) = s/N$ 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 6 shows the predicted probabilities on the SNLI test data when our model is trained on different distributions of data. The top row shows the predictions of our model when trained only on short phrases from the denotation graph. We observe that the median probabilities increase from contradiction to neutral to entailment, even though this model was only trained on short phrases with a limited vocabulary. Given the training data, we did not expect these probabilities to align cleanly with the entailment labels, but even so, there is already some information here to distinguish between entailment classes.

The bottom row shows that when our model is trained on both denotational phrases and SNLI sentence pairs with approximate conditional probabilities, its probability predictions for longer sentences improve. This model's predicted conditional probabilities align much more closely with the entailment class labels. Entailing sentence pairs have high conditional probabilities (median 0.72), neutral sentence pairs have mid-range conditional probabilities (median 0.46), and contradictory sentence pairs have conditional probabilities approaching 0 (median 0.19).

8 Predicting textual entailment

In Section 7.2, we trained our probability model on both short phrase pairs for which we had gold probabilities and longer SNLI sentence pairs for which we estimated probabilities. We now evaluate the effectiveness of this model for textual entailment, and demonstrate that these predicted probabilities are informative features for predicting entailment on both SICK and SNLI.

Model We first train an LSTM similar to the 100d LSTM that achieved the best accuracy of the neural models in Bowman et al. (2015). It takes GloVe word vectors as input and produces 100d sentence vectors for the premise and hypothesis. The concatenated 200d sentence pair representation from the LSTM passes through three 200d $\tanh$ layers and a softmax layer for 3-class entailment classification. We train the LSTM on the SNLI training data with batch size 512 for 10 epochs. We use the Adam optimizer with a learning rate of 0.001 and a dropout rate of 0.85, and use the development data to select the best model.

Model	Test Acc.
Our LSTM	77.2
Our LSTM + CPR	**78.2**
Bowman et al. (2015) LSTM	77.2

Table 2: Entailment accuracy on SNLI (test).

Model	Test Acc.
Our LSTM	81.5
Our LSTM + CPR	**82.7**
Bowman et al. (2015) transfer	80.8

Table 3: Entailment accuracy on SICK (test).

Next, we take the output vector produced by the LSTM for each sentence pair and append our predicted $P_{[\![]\!]}(h|p)$ value (the probability of the hypothesis given the premise). We train another classifier that passes this 201d vector through two $\tanh$ layers with a dropout rate of 0.5 and a final 3-class softmax classification layer. Holding the parameters of the LSTM fixed, we train this model for 10 epochs on the SNLI training data with batch size 512.

Results Table 2 contains our results on SNLI. Our baseline LSTM achieves the same 77.2% accuracy reported by Bowman et al. (2015), whereas a classifier that combines the output of this LSTM with only a single feature from the output of our probability model improves to 78.2% accuracy.

We use the same approach to evaluate the effectiveness of our predictions on SICK (Table 3). SICK does not have enough data to train an LSTM, so we combine the SICK and SNLI training data to train both the LSTM and the final model. When we add the predicted conditional probability as a single feature for each SICK sentence pair, performance increases from 81.5% to 82.7% accuracy. This approach outperforms the transfer learning approach of Bowman et al. (2015), which was also trained on both SICK and SNLI.

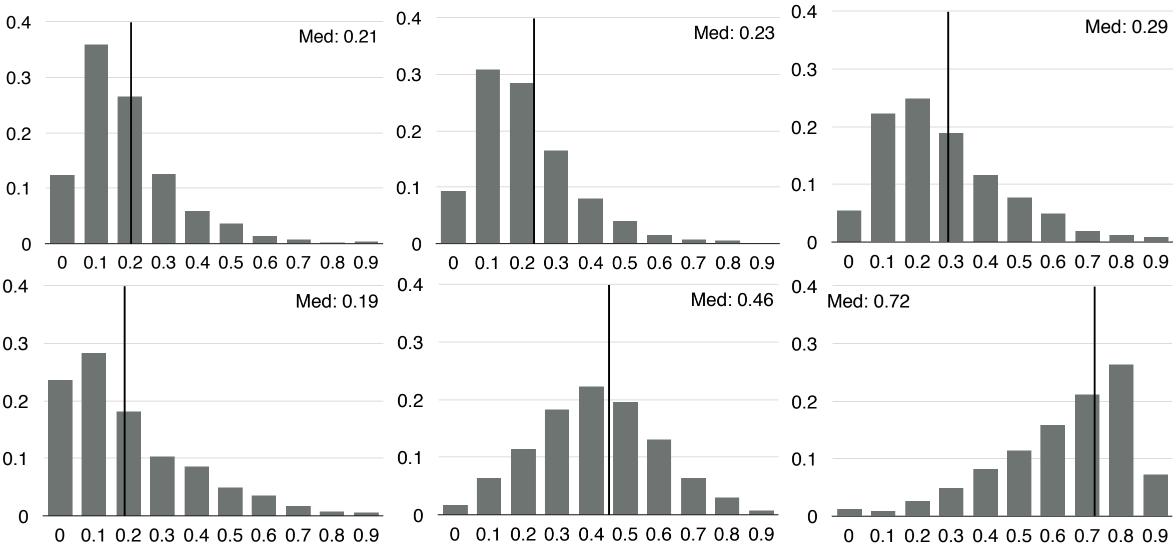


Figure 6: Predicted conditional probabilities $P(h|p)$ for SNLI sentence pairs (test) by entailment label, as a percentage of pairs with that label. Top: predictions from the model trained only on short denotational phrases. Bottom: predictions from the model trained on both short denotational phrases and SNLI.

	Premise	Hypothesis	G	P
1	person walk on trail in woods	in forest	1.0	1.0
2	group of person bike	group of person ride	0.9	0.8
3	adult sing while play instrument	adult play guitar	0.8	0.8
4	person sit on bench outside	on park bench	0.4	0.4
5	tennis player hit ball	person swing	0.2	0.2
6	girl sleep	on pillow	0.1	0.2
7	man practice martial art	person kick person	0.1	0.3
8	person skateboard on ramp	man ride skateboard	0.2	0.2
9	busy intersection	city street	0.3	0.2
10	person dive into swim pool	person fly through air	0.1	0.1
11	sit at bench	adult read book	0.1	0.1
12	person leap into air	jump over obstacle	0.0	0.0
13	person talk on phone	man ride skateboard	0.0	0.0

Table 4: Gold and predicted conditional probabilities from the denotational phrase development data.

Premise	Hypothesis	Gold	Pred
skier on snowy hill	athlete	1.00	0.99
pitcher throw ball	mound	0.53	0.84
golf ball	athlete	0.53	0.66
person point	man point	0.48	0.41
in front of computer	person look	0.36	0.21

Table 5: Gold and predicted conditional probabilities from unseen pairs in the denotational phrase development data.

9 Discussion

Section 7 has demonstrated that we can successfully learn to predict denotational probabilities for phrases that we have not encountered during training and for longer sentences. Section 8 has illustrated the utility of these probabilities by showing that a single feature based on our model's predicted conditional denotational probabilities improves the accuracy of an LSTM on SICK and SNLI by 1 percentage point or more. Although we were not able to 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 6 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).

We examine examples of predicted conditional probabilities for phrase and sentence pairs to analyze our model's strengths and weaknesses. Table 4 has example predictions from the denotation phrase development data. Our model correctly predicts high conditional probability for entailed phrase pairs even when there is no direct hypernym involved, as in example 2, and for closely related phrases that are not strictly entailing, as in example 3. Our model also predicts reasonable probabilities for events that frequently co-occur but are not required to do so, such as example 7.

		Premise	Hypothesis	CPR
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 6: Predicted conditional probabilities for sentence pairs from the SNLI development data.

In examples 10 and 11, our model predicts low probabilities for occasionally co-occurring events, which are still more likely than the improbable cooccurrence in example 13. Table 5 demonstrates similar patterns for pairs where both phrases were unseen.

Table 6 has examples of predicted conditional probabilities for sentence pairs from the SNLI development data. Some cases of entailment are straightforward, so predicting high conditional probability is relatively easy. This is the case with example 2, which simply involves dropping words from the premise to reach the hypothesis. In other cases, our model correctly predicts high conditional probability for an entailed hypothesis that does not have such obvious word-to-word correspondence with the premise, such as example 1. Our model's 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. For neutral pairs, our model usually predicts midrange probabilities, but there are some exceptions. In example 6, it is not certain that the people are playing music, but it is a reasonable assumption from the premise. It makes sense that in this case, our model assigns this hypothesis a higher conditional probability given the premise than for most neutral sentence pairs. In example 7, we might guess that the patient is a man with 50% probability, so the predicted conditional probability of our model seems reasonable. Our model cannot reason about numbers and quantities, as example 10 shows. It also fails to predict in example 11 that a man wearing a black shirt is probably not wearing a blue shirt as well. However, our model does correctly predict low probabilities for some contradictory examples that have reasonably high word overlap, as in example 13. Finally, example 14 shows that our model can correctly predict very low conditional probability for sentences that share no common subject matter.

10 Conclusion

We have presented a framework for representing denotational probabilities in a vector space, and demonstrated that we can successfully train a neural network model to predict these probabilities for new phrases. We have shown that when also trained on longer sentences with approximate probabilities, our model can learn reasonable representations for these longer sentences. We have also shown that our model's predicted probabilities are useful for textual entailment, and provide additional gains in performance when added to existing competitive textual entailment classifiers. Future work will examine whether the embeddings our model learns can be used directly by these classifiers, and explore how to incorporate negation into our model.

Acknowledgments

This work was supported by NSF Grants 1563727, 1405883, and 1053856, and by a Google Research Award. Additional thanks to Yonatan Bisk and Pooya Khorrami.

References

Islam Beltagy, Stephen Roller, Pengxiang Cheng, Katrin Erk, and Raymond J. Mooney. 2015. Representing meaning with a combination of logical form and vectors. *CoRR*, abs/1505.06816.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2012. Learning entailment relations by global graph structure optimization. *Computational Linguistics*, 38(1):73–111, March.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, and Hui Jiang. 2016. Enhancing and combining sequential and tree LSTM for natural language inference. *CoRR*, abs/1609.06038.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, MLCW'05, pages 177–190, Berlin, Heidelberg. Springer-Verlag.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. Recognizing textual entailment: Models and applications. *Synthesis Lectures on Human Language Technologies*, 6(4):1–220.

Katrin Erk. 2009. Representing words as regions in vector space. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 57–65, Boulder, Colorado, June.

James Henderson and Diana Popa. 2016. 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)*, pages 2052–2062, Berlin, Germany, August.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November.

Germn Kruszewski, Denis Paperno, and Marco Baroni. 2015. Deriving boolean structures from distributional vectors. *Transactions of the Association for Computational Linguistics*, 3:375–388.

Alice Lai and Julia Hockenmaier. 2014. Illinois-LH: A denotational and distributional approach to semantics. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 329–334.

Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016. Learning natural language inference using bidirectional LSTM model and inner-attention. *CoRR*, abs/1605.09090.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. 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)*.

Tsendsuren Munkhdalai and Hong Yu. 2017a. Neural semantic encoders. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*.

Tsendsuren Munkhdalai and Hong Yu. 2017b. Neural tree indexers for text understanding. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas, November.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomas Kocisky, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In *The International Conference on Learning Representations (ICLR)*.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2016. Order-embeddings of images and language. In *The International Conference on Learning Representations (ICLR)*.

Luke Vilnis and Andrew McCallum. 2015. Word representations via gaussian embedding. In *The International Conference on Learning Representations (ICLR)*.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations. *Transactions of the Association of Computational Linguistics – Volume 2, Issue 1*, pages 67–78.