Learning Distributed Word Representations for Natural Logic Reasoning

Samuel R. Bowman, Christopher Potts, Christopher D. Manning
Stanford University
450 Serra Mall
Stanford, CA 94305-2150
{sbowman,cgpotts,manning}@stanford.edu

Abstract

Natural logic offers a powerful relational conception of meaning that is a natural counterpart to distributed semantic representations, which have proven valuable in a wide range of sophisticated language tasks. However, it remains an open question whether it is possible to train distributed representations to support the rich, diverse logical reasoning captured by natural logic. We address this question using two neural network-based models for learning embeddings: plain neural networks and neural tensor networks. Our experiments evaluate the models’ ability to learn the basic algebra of natural logic relations from simulated data and from the WordNet noun graph. The overall positive results are promising for the future of learned distributed representations in the applied modeling of logical semantics.

Natural logic offers a powerful relational conception of semantics: the meanings for expressions are given, at least in part, by their inferential connections with other expressions (Katz 1972; MacCartney and Manning 2009; van Benthem 2008). For instance, turtle is analyzed, not primarily by its extension in the world, but rather by its lexical network: it entails reptile, excludes chair, is entailed by sea turtle, and so forth. With generalized notions of entailment and contradiction, these relationships can be defined for all lexical categories as well as complex phrases, sentences, and even texts. The resulting theories of meaning offer valuable new analytic tools for tasks involving database inference, relation extraction, and textual entailment.

Natural logic aligns well with distributed (e.g., vector) representations, which also naturally model meaning relationally. Distributed representations have been used successfully in a wide array of sophisticated language tasks, including part of speech tagging, (Collobert et al. 2011), semantic role labeling (Collobert et al. 2011), sentiment analysis (Socher et al. 2011), and translation (Sutskever, Vinyals, and Le 2014) among many others. However, it remains an open question whether it is possible to train such representations to support the rich, diverse logical reasoning captured by natural logic: while they excel at synonymy (similarity), the results are more mixed for entailment, contradiction, and mutual consistency. Using the natural logic of (MacCartney and Manning 2009) as our formal model, we address this open question using two neural network-based models for learning embeddings: plain neural networks and neural tensor networks (NTNs). The natural logic is built from the seven relations defined in Table 1. Its formal properties are now well-understood (Icard and Moss 2013a; 2013b), so it provides a rigorous set of goals for our neural models. To keep the discussion manageable, we limit attention to experiments involving the lexicon; for a more extended treatment of complex expressions involving logical connectives and quantifiers, see (Bowman, Potts, and Manning 2014).

In our experiments, we evaluate these models’ ability to learn the basic algebra of natural logic relations from simulated data and from the WordNet noun graph. The simulated data help us to achieve analytic insights into what the models learn, and the WordNet data show how they fare with a real natural language vocabulary. We find that only the NTN is able to fully learn the underlying algebra, but that both models excel in the WordNet experiment.

1 Neural network models for relation classification

We build embedding-based models using the method of (Bowman, Potts, and Manning 2014), which is centered on the task of labeling a pair of words or sentences with one of a small set of logical relations. Unlike in a classical inference setting, the model is not given specific premises to reason with at test time, but rather is expected to encode all of the information that it learns about each term (word) at training time in the term embedding vectors, and to then use only the information encoded in the relevant pair of term vectors at test time. The architecture that we use is depicted in Figure 1. The model represents the two input terms as single embedding vectors, which are fed into a comparison function based on one of two types of neural network layer functions to produce a representation for the relationship between the two terms. This representation is then fed into a softmax classifier, which outputs a distribution over possible labels. The entire network, including the embeddings, is trained using backpropagation with AdaGrad (Duchi, Hazan, and Singer 2011) and L2 regularization.

The simpler version of the comparison concatenates the
For our experiment on WordNet data, we introduce an additional third-order tensor parameter to allow for multiplicative interactions between the two inputs (Chen et al. 2013). For more details on the implementation and training of the NTN model, see (Bowman, Potts, and Manning 2014).

This model used here differs from the one described in (Hannun, and Ng 2013) neural network (NN) layer. The more powerful neural tensor network (NTN) version adds an additional third-order tensor parameter to allow for multiplicative interactions between the two inputs. For more details on the implementation and training of the layer functions, see (Bowman, Potts, and Manning 2014).

In this experiment, we test our model's ability to learn these inferential patterns, we create artificial boolean structures in which terms denote sets of entities from a small domain (e.g., Figure 2b), employ logical deduction to identify the valid statements, divide those into train and test sets, and remove from the test set those statements which cannot be proven from the training statements (Figure 2c). In our experiments, we create 80 randomly generated sets drawn from a domain of seven elements. This yields 6400 statements about pairs of formulae, of which 3200 are chosen as a test set. 240 of the test examples were excluded from the primary test set for lack of evidence in the training data to support the choice of any one relation. We then trained and evaluated both the NN model and the NTN model on these data sets.

Results We found that the NTN model worked best with 11-dimensional vector representations for the 80 sets and a 90-dimensional feature vector for the classifier, although the performance of the model was not highly dependant on either dimensionality setting. Averaging over five randomly generated data sets, the model was able to correctly label 98.1% (standard error 0.67%) of the provably derivable test examples, and 87.7% (SE = 3.59%) of the remaining test examples. The simpler NN worked best with 11 and 75 dimensions, respectively, but was able to achieve accuracies of only 84.8% (SE = 0.84%) and 68.7% (SE = 1.58%), respectively. Training accuracy was 99.8% (SE = 0.04%) for the NTN and 94.7% (SE = 0.89%) for the NN.

These results are fairly straightforward to interpret. The NTN model was able to accurately encode the relations between the terms in the geometric relations between their vectors, and was able to then use that information to recover relations that were not overtly included in the training data. In contrast, the NN was able to achieve this behavior only incompletely. It is possible but not likely that it could be made to find a good solution with further optimization on different learning algorithms, or that it would do better on a larger universe of sets, for which there would be a larger set of training data to learn from, but the NTN is readily able to achieve these effects in the setting discussed here.

It’s noteworthy that both models performed well above chance on the unprovable examples. This indicates that both

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Set-theoretic definition</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>entailment</td>
<td>$x \sqsubseteq y$</td>
<td>$x \subseteq y$</td>
<td>turtle $\sqsubseteq$ reptile</td>
</tr>
<tr>
<td>reverse entailment</td>
<td>$x \sqsupset y$</td>
<td>$x \supset y$</td>
<td>reptile $\supset$ turtle</td>
</tr>
<tr>
<td>equivalence</td>
<td>$x \equiv y$</td>
<td>$x = y$</td>
<td>couch $\equiv$ sofa</td>
</tr>
<tr>
<td>alternation</td>
<td>$x</td>
<td>y$</td>
<td>$x \cap y = \emptyset \land x \cup y \neq D$</td>
</tr>
<tr>
<td>negation</td>
<td>$x \wedge y$</td>
<td>$x \cap y = \emptyset \lor x \cup y = D$</td>
<td>able $\wedge$ unable</td>
</tr>
<tr>
<td>cover</td>
<td>$x \sim y$</td>
<td>$x \cap y \neq \emptyset \land x \cup y = D$</td>
<td>animal $\sim$ non-turtle</td>
</tr>
<tr>
<td>independence</td>
<td>$x # y$</td>
<td>(else)</td>
<td>turtle $#$ pet</td>
</tr>
</tbody>
</table>

Table 1: The seven natural logic relations of (MacCartney and Manning 2009). $D$ is the universe of possible objects of the same type as those being compared, and the relation $\#$ applies whenever none of the other six do.
were able to identify statistical cues to the properties of the underlying boolean structure, thereby going beyond what one could obtain from the natural logic. Information of this kind is not sufficient to yield the near-perfect accuracy that we saw on the provable data, but it does highlight one of the ways in which trained inference models can turn out to be more powerful that rigidly defined proof systems.

3 Reasoning about lexical relations in WordNet

Using simulated data as above is reassuring about what the models learn and why, but we also want to know how they perform with a real natural language vocabulary. Unfortunately, as far as we are aware, there are no available resources labeling such a vocabulary with the relations from Table 1. However, the relations in WordNet (Fellbaum 2010; Miller 1995) come close and pose the same substantive challenges within a somewhat easier classification problem.

We extract three types of relation from WordNet. Hypernym and hyponym can be represented directly in the WordNet graph structure, and correspond closely to the $\equiv$ and $\sqsubset$ relations from natural logic. As in natural logic, these relations are mirror images of one another: if $\text{dog}$ is a hypernym of $\text{animal}$ (perhaps indirectly by way of $\text{canid}$, mammal, etc.), then $\text{animal}$ is a hypernym of $\text{dog}$. We also extract coordinate terms, which share a direct hypernym, like $\text{dalmatian}$, $\text{pug}$, and $\text{puppy}$, which are all direct hypernyms of $\text{dog}$. Coordinate terms tend to exclude one another, thereby providing a loose approximation of the natural logic exclusion relation $\setminus$. WordNet’s antonym relation, however, is both too narrowly defined and too rare to use for this purpose. WordNet defines its relations over sets of synonyms, rather than over individual terms, so we do not include a synonym or equivalent relation, but rather consider only one member of each set of synonyms. Word pairs which do not fall into these three relations are not included in the data set.

To limit the size of the vocabulary without otherwise simplifying the learning problem, we extract all of the instances of these three relations for single word nouns in WordNet that are hypernyms of the node $\text{organism.n.01}$. In order to balance the distribution of the classes, we slightly down-sample instances of the coordinate relation, yielding a total of 36,772 relations among 3,217 terms. We report results below using cross-validation, choosing a disjoint 10% test sample for each of five runs. Unlike in the previous experiment, it is not straightforward here to determine in advance how much data should be required to train an accurate model, so we performed training runs with various fractions of the remaining data. Embeddings were fixed at 25 dimensions and were initialized randomly or using distributional vectors from GloVe (Pennington, Socher, and Manning 2014). The feature vector produced by the comparison layer was fixed at 80 dimensions.

Results The structured WordNet data contains substantially more redundancy than the random artificial data above, and we find that the NTN performs perfectly with random initialization. The plain NN performs almost as well, providing a point of contrast with the results of Section 2. We also find that initialization with GloVe is helpful in allowing the models to maintain fair performance with smaller amounts of training data. Some of the randomly initialized model runs failed to learn usable representations at all and labeled all examples with the most frequent labels. We excluded these runs from the statistics, but marked settings for which this occurred with the symbol $\dagger$. For all of the remaining runs, training accuracy was consistently above 99%.

4 Conclusion

This investigation builds upon extensive prior research on constructing compositional word meanings in vector spaces (Clark, Coecke, and Sadrzadeh 2011; Grefenstette 2013; Hermann, Grefenstette, and Blunsom 2013; Rocktäschel et al. 2014), extracting entailment information from distributional vector space models (Baroni et al. 2012; Rei and Briscoe 2014), and transferring information between logic programs and neural networks in constrained settings (Hitzer, Hölldobler, and Seda 2004; Hölldobler, Kalinke, and Störr 1999). Our primary contribution is to establish a tight
connection between these approaches and formal developments within natural logic. More specifically, we showed that neural network models, optimized using established techniques, can learn distributed representations that closely approximate the behavior one would expect from a natural logic inference algorithm.

This paper evaluated two neural models on the task of learning natural logic relations between distributed word representations. The results suggest that at least the neural tensor network has the capacity to meet this challenge with reasonably-sized training sets, learning both to embed a vocabulary in a way that encodes a diverse set of relations, and to subsequently use those embeddings to infer new relations. In (Bowman, Potts, and Manning 2014), we extend these results to include complex expressions involving logical connectives and quantifiers, with similar conclusions about (tree-structured recursive versions of) these models. These findings are promising for the future of learned distributed representations in the applied modeling of logical semantics.

5 Acknowledgments

We thank Jeffrey Pennington, Richard Socher, and audiences at CSLI, Nuance, and BayLearn.

References


Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. JMLR.


