Learning Distributed Word Representations for Natural Logic Reasoning

Samuel R. Bowman, Christopher Potts, and 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 [14, 16, 23]. 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 speach tagging, [5], semantic role labeling [5], sentiment analysis [21], and translation [22] 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 [16] 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 [12, 13], 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 [2].

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.

Copyright © 2015, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

1 Neural network models for relation classification

We build embedding-based models using the method of [2], 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 function 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 [6] and L2 regularization.

The simpler version of the comparison concatenates the two input vectors before feeding them into a LReLU [15] 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 [3]. For more details on the implementation and training of the layer functions, see [2].

Name	Symbol	Set-theoretic definition	Example
entailment	$x \sqsubset y$	$x \subset y$	*turtle* $\sqsubset$ *reptile*
reverse entailment	$x \sqsupset y$	$x \supset y$	*reptile* $\sqsupset$ *turtle*
equivalence	$x \equiv y$	$x = y$	*couch* $\equiv$ *sofa*
alternation	$x \mid y$	$x \cap y = \emptyset \wedge x \cup y \neq \mathcal{D}$	*turtle* $\mid$ *warthog*
negation	$x \wedge y$	$x \cap y = \emptyset \wedge x \cup y = \mathcal{D}$	*able* $\wedge$ *unable*
cover	$x \smile y$	$x \cap y \neq \emptyset \wedge x \cup y = \mathcal{D}$	*animal* $\smile$ *non-turtle*
independence	$x \# y$	(else)	*turtle* $\#$ *pet*

Table 1: The seven natural logic relations of [16]. $\mathcal{D}$ 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.

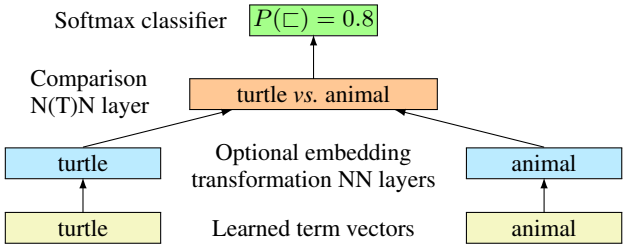


Figure 1: The model structure used to compare *turtle* and *animal*. Learned term representations are fed through either an NN or NTN comparison layer and then to a softmax classifier over the seven relations in Table 1.

This model used here differs from the one described in that work in two ways. First, because the inputs are single terms, we do not use a composition function here. Second, for our experiment on WordNet data, we introduce an additional $\tanh$ neural network layer between the embedding input and the comparison function, which facilitates initializing the embeddings themselves from pretrained vectors and was found to help performance in that setting.

2 Reasoning about semantic relations

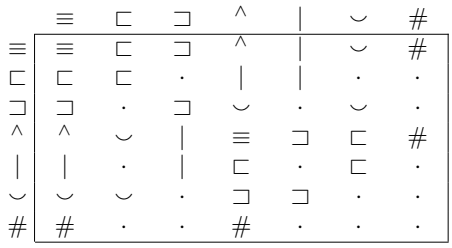
In this experiment, we test our model's ability to learn and use the natural logic inference rules schematized in Figure 2a. For instance, given that $a \sqsupset b$ and $b \sqsupset c$, one can conclude that $a \sqsupset c$, by basic set-theoretic reasoning (transitivity of $\sqsupset$). Similarly, from $a \sqsubset b$ and $b \wedge c$, it follows that $a \mid c$. There are 32 valid inferences that can be made on the basis of pairs of relations of the form aRb and $bR'c$. Cells in the table containing a dot correspond to pairs of relations for which no valid inference can be drawn in our logic.

Experiments To test our models' 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.

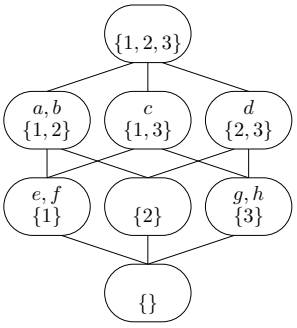
Results We found that the NTN model worked best with 11-dimensional vector representations for the 80 sets and a 90-dimensional feature vector for the classifier, though the performance of the model was not highly dependant on either dimensionality setting. Averaging over five randomly generated data 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 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.



(a) Inference path from premises $a\,R\,b$ (row) and $b\,S\,c$ (column) to the relation that holds between a and c, if any. These inferences are based on basic set-theoretic truths about the meanings of the underlying relations as described in Table 1. We assess our models' ability to reproduce such inferential paths.



(b) Simple boolean structure. The letters name the sets. Not all sets have names, and some sets have multiple names, so that learning ≡ is non-trivial.

Train	Test
$a \equiv b$	$b \equiv b$
$a \sqsubset e$	$b \sqsubset e$
$a \wedge g$	~~$e \equiv f$~~
$d \sqsubset h$	~~$g \sqsupset d$~~
$e \mid g$	$h \sqsupset d$
⋮	⋮

(c) A train/test split of the atomic statements about the model. Test statements not provable from the training data are crossed out.

Figure 2: Experimental goal and set-up for reasoning about semantic relations.

Portion of training data	25d NN w/ GloVe	25d NN w/o GloVe	25d NTN w/ GloVe	25d NTN w/o GloVe	⊏ only
100%	99.73	99.37	99.61	**99.95**	37.05
33%	95.96	95.30	95.83	95.45	37.05
11%	91.11	90.81	91.27	90.90	37.05

Table 2: Mean test % accuracy scores (with standard error) on the WordNet data over five-fold crossvalidation. The baseline figure is the frequency of the most frequent class, *hypernym*.

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 [7, 17] 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 ⊐ and ⊏ relations from natural logic. As in natural logic, these relations are mirror images of one another: if *dog* is a hyponym of *animal* (perhaps indirectly by way of *canid*, *mammal*, etc.), then *animal* is a hypernym of *dog*. We also extract *coordinate* terms, which share a direct hypernym, like *dalmatian*, *pug*, and *puppy*, which are all direct hyponyms of *dog*. Coordinate terms tend to exclude one another, thereby providing a loose approximation of the natural logic exclusion relation |. 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 hyponyms of the node `organism.n.01`. In order to balance the distribution of the classes, we slightly downsample instances of the *coordinate* relation, yielding a total of 36,772 relations among 3,217 terms. We report results below using crossvalidation, 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 [18]. 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 †. 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 [4, 8, 9, 20], extracting entailment information from distributional vector space models [1, 19], and transferring information between logic programs and neural networks in constrained settings [10, 11]. 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 [2], 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

[1] Baroni, M.; Bernardi, R.; Do, N.-Q.; and Shan, C.-c. 2012. Entailment above the word level in distributional semantics. In *Proc. EACL*.

[2] Bowman, S.; Potts, C.; and Manning, C. 2014. Recursive neural networks can learn logical semantics. arXiv manuscript 1406.1827.

[3] Chen, D.; Socher, R.; Manning, C.; and Ng, A. 2013. Learning new facts from knowledge bases with neural tensor networks and semantic word vectors. In *Proc. ICLR*.

[4] Clark, S.; Coecke, B.; and Sadrzadeh, M. 2011. Mathematical foundations for a compositional distributed model of meaning. *Linguistic Analysis* 36(1–4):345–384.

[5] Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; and Kuksa, P. 2011. Natural language processing (almost) from scratch. *JLMR* 12.

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

[7] Fellbaum, C. 2010. WordNet. In *Theory and Applications of Ontology: Computer Applications*. Springer.

[8] Grefenstette, E. 2013. Towards a formal distributional semantics: Simulating logical calculi with tensors. arXiv manuscript 1304.5823.

[9] Hermann, K.; Grefenstette, E.; and Blunsom, P. 2013. "Not not bad" is not "bad": A distributional account of negation. In *Proc. of the 2013 Workshop on Continuous Vector Space Models and their Compositionality*.

[10] Hitzler, P.; Hölldobler, S.; and Seda, A. K. 2004. Logic programs and connectionist networks. *Journal of Applied Logic* 2(3).

[11] Hölldobler, S.; Kalinke, Y.; and Störr, H.-P. 1999. Approximating the semantics of logic programs by recurrent neural networks. *Applied Intelligence* 11(1).

[12] Icard, T., and Moss, L. 2013a. A complete calculus of monotone and antitone higher-order functions. In *Proc. Topology, Algebra, and Categories in Logic*.

[13] Icard, T., and Moss, L. 2013b. Recent progress on monotonicity. *Linguistic Issues in Language Technology* 9(7).

[14] Katz, J. J. 1972. *Semantic Theory*. New York: Harper & Row.

[15] Maas, A.; Hannun, A.; and Ng, A. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML*.

[16] MacCartney, B., and Manning, C. 2009. An extended model of natural logic. In *Proc. IWCS*.

[17] Miller, G. A. 1995. WordNet: A lexical database for English. *Communications of the ACM* 38(11):39–41.

[18] Pennington, J.; Socher, R.; and Manning, C. 2014. GloVe: Global vectors for word representation. In *Proc. EMNLP*.

[19] Rei, M., and Briscoe, T. 2014. Looking for hyponyms in vector space. In *Proc. CoNLL*.

[20] Rocktäschel, T.; Bosnjak, M.; Singh, S.; and Riedel, S. 2014. Low-dimensional embeddings of logic. In *Proc. the ACL 2014 Workshop on Semantic Parsing*.

[21] Socher, R.; Pennington, J.; Huang, E.; Ng, A.; and Manning, C. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proc. EMNLP*.

[22] Sutskever, I.; Vinyals, O.; and Le, Q. 2014. Sequence to sequence learning with neural networks. In *Proc. NIPS*.

[23] van Benthem, J. 2008. A brief history of natural logic. In Chakraborty, M.; Löwe, B.; Nath Mitra, M.; and Sarukki, S., eds., *Logic, Navya-Nyaya and Applications: Homage to Bimal Matilal*.