Logical Reasoning in Compositional Vector Space Semantics

Samuel R. Bowman
LING 236

June 2, 2013

1 Introduction

Vector space models of semantic meaning have substantially advanced the state of the art in applied natural language processing tasks like parsing, paraphrase detection, and sentiment analysis. These models have benefited in particular from two properties: The ability to learn representations of individual words that are *optimized for the task at hand*, and their ability to compose those word representations into higher-order constituent representations *in the same semantic vector space*. Together, these properties allow the model to capture nuanced compositional representations of sentence meaning without the need for any independent lexical semantic model.

There is not at this time any general understanding of what *kind* of information these models are capable of learning and encoding in the vector space. It is clear that they are able to learn various useful facets of meaning well, but it is possible that many of the properties that are necessary for a full account of linguistic meaning like quantification and monotonicity are missed: while these models achieve state-of-the-art performance, the models that they supplant are largely models based on simple expert-designed features with substantial known limitations.

In this study, I propose to probe the representational capacities of these systems. In particular, I propose to build a model which learns task-specific compositional vector representations—in the style of Socher et al.—for the task of recognizing textual entailment in a regimented formal language. This model should shed light on the abilities of these schemes to capture reasoning patterns involving quantification, monotonicity, intersective and nonintersective modification, and other phenomena.

2 The Pilot Experiment: Reasoning with idealized data

If the pilot model works: Distributional system with X composition function gets quantification. Publish and push on how best to learn it in broader domain applications.

If it fails: Can we say that the system isn't powerful enough? Squib and return to even more supervised approaches.

3 Background

Baroni et al. (2012) present a real world version of the entailment reasoning problem, and have the current state of the art on that task. [Ken Shan is the last author on this.]

> Interestingly, on the QN entailment task, neither our classifier trained on AN-N pairs nor the balAPinc method beat baseline methods. This suggests that our successful QN classifiers tap into vector properties beyond such relations as feature inclusion that those methods for nominal entailment rely upon.
>
> The FS definition of entailment has been modified by taking common sense into account. Instead of a relation from the truth of the consequent to the truth of the antecedent in any circumstance, the applied view looks at entailment in terms of plausibility
>
> ... if a human who reads (and trusts) would most likely infer that is also true.

Erk (2009) discusses notions of subsumption from distributional vectors. His results would be discouraging if we were trying to use pure distributional vectors rather than task-learned ones.

> *Baroni et al on Erk:* Erk (2009) suggests that it may not be possible to induce lexical entailment directly from a vector space representation, but it is possible to encode the relation in this space after it has been derived through other means.
>
> ...run implies move in some contexts, but not all, for example not in the context computer runs.

Baroni and Lenci (2011) present a corpus of word pairs manually annotated with relations like *hypernym*, *hyponym*, *meronym*, *associated event*, and *common attribute*, as well as negative example relations.

```
missile-n weapon hyper rocket-n
missile-n weapon hyper vehicle-n
missile-n weapon hyper weapon-n
missile-n weapon mero engine-n
missile-n weapon mero explosive-n
```

This may be useful as a fringe test case, but the lack of proper entailment/cover/equiv relationships may make it more difficult to test the logical capacity of the model by these means.

Lenci and Benotto (2012) have more of the same: distributional inclusion does capture some entailment information.

> We introduce BLESS, a data set specifically designed for the evaluation of distributional semantic models. BLESS contains a set of tuples instantiating different,

explicitly typed semantic relations, plus a number of controlled random tuples. It is thus possible to assess the ability of a model to detect truly related word pairs, as well as to perform in-depth analyses of the types of semantic relations that a model favors. We discuss the motivations for BLESS, describe its construction and structure, and present examples of its usage in the evaluation of distributional semantic models.

In my notes for some reason: LDC's ACE08 Local Relation Detection and Recognition (LRDR) task

Goller and Kuchler (1996) present the core of the machine learning algorithm needed for the task-specific representations used in this project. I can't make heads or tails of their writing.

Widdows's (2004) book primarily serves as an primer on vector space semantics for those without sufficient mathematical background, but it does briefly address some aspects of the logical capacities of such systems. In particular, he presents a system for constructing information-retrieval targeted query vectors, and builds a practical quasi-logic around it which is capable of conjunction (*rock* AND *blues*) using vector addition, and subtraction (*rock* NOT *blues*) through the the operation of finding the orthogonal vector, expressed as:

(1) $a \text{ NOT } b = a - (a \cdot b)b$

MacCartney and Manning (2009) (from Bill's dissertation Maccartney, 2009) presents the framing of entailment and monotonicity used here, and shows that they can (with a lot of hacks) be used to do the Recognizing Textual Entailment task on real language. No vectors here.

Icard III (2012) presents an extension to the NLI/MacCartney and Manning logic involving an elaborated notion of monotonicity. He uses the same seven basic entailment relations as MacCartney. The added detail relates to the model described in this paper as a formal description of what the model should ideally be able to learn, and not as a description of a formal property that should be externally introduced into the model.

Socher et al. (2011a) is the primary introduction to the idea of learning a set of quasi-distributional vectors and a composition function on those vectors, all guided to optimize performance on some task.

Socher et al. (2011b) extends this to paraphrase detection, an (easy, by our standards) task of comparing two learned sentence representations.

Socher et al. (in prep; ACL 2013) and Chen et al. (2013) define a more powerful version of the composition function based on a third-order tensor. I'm using this layer as the top node that builds the comparison vector from which relation types are extracted.

Socher and co. have a few other recent and in-progress papers setting various task-performance records. Useful for motivation, but not that informative.

Basile et al: Encoding syntactic dependencies by vector permutation. Slides are a bit vague. Give it a closer look!

4 Methods

4.1 Forward computation: Identifying a single relation

4.1.1 The probability function

The top layer of the model is a softmax classifier, which takes the model output and transforms it into a multinomial probability distribution over the seven possible relation types

This takes the input from the layer below, $\vec{y}$, appends a 1 to yield a bias term, multiplies the result by a matrix $\mathbf{S}$ and exponentiates it elementwise to yield an unnormalized probability measure.

(2) $\vec{\mu}(\vec{y}) = exp(\mathbf{S} \cdot [1; \vec{y}])$

From this, we can normalize to get a probability function by dividing each entry in the vector by the sum of the vector:

(3) $\vec{\pi}(\vec{y}) = \dfrac{\vec{\mu}(\vec{y})_r}{\sum_{r'=1}^{R} \mu(\vec{y})_{r'}}$

(4) $P(rel(x_1, x_2) = r) = \pi(\vec{y})_r$

$\mathbf{S}$ above is a $D' \times R$ matrix.

4.1.2 The comparison function

The next layer is the comparison layer. This takes in the two D-dimensional representations of each constituent, $\vec{x^1}$ and $\vec{x^2}$, and transform them into a D'-dimensional vector y representing the relation between the two. While this layer is not intended to be the same function as the composition function that builds up the individual constituents, I parameterize it in the same way, as an RNTN layer with three parameters, one a third-order tensor (in other words, a vector of matrices, marked $\mathbf{A}^{[\mathbf{1} \ldots \mathbf{D'}]}$), one a matrix ($\mathbf{B}$), and one a vector ($\vec{c}$).

Each element in the output vector is computed using a single matrix drawn from the third order tensor parameter, a single row of the matrix parameter, and a single entry of the vector parameter, as follows.

(5) $y_i = f(\vec{x^1}^T \mathbf{A}^{[\mathbf{1} \ldots \mathbf{D'}]} \vec{x^2} + \vec{B_{i,:}}[\vec{x^1}; \vec{x^2}] + c_i)$

(6) $Cpr(\vec{x^1}, \vec{x^2}) = [y_1; y_2; \ldots; y'_D]$

The result of each of these computations is fed into a sigmoid nonlinearity f. In my implementation, I use $f(x) = \tanh(x)$.

4.1.3 The composition function

As with the comparison layer, the composition layer is an RNTN layer with three parameters, one a third-order tensor ($\mathbf{K}^{[\mathbf{1}...\mathbf{D}']}$), one a matrix ($\mathbf{L}$), and one a vector ($\vec{m}$). As above, the output of the layer is computed as follows.

(7) $p_i = f(\vec{x^1}^T \mathbf{K}^{[\mathbf{1}...\mathbf{D}]}\vec{x^2} + \vec{L_{i,:}}[\vec{x^1};\vec{x^2}] + m_i)$

(8) $Mrg(\vec{x^1},\vec{x^2}) = [y_1; y_2; ...; y'_D]$

4.2 Backpropagation: Learning from a single relation

4.2.1 The Objective

For an objective function, I use the negative logarithm of the probability of the true relation to capture correctness, and the sum of the squared parameter values (L2 normalization) to minimize overfitting. I present it here as a function of the two top-level vectors of the inputs, and the single correct relation specification.

(9) $E(\vec{w_1},\vec{w_2},\vec{r},\theta) = -\sum_{i=1}^{N} \log(\pi(Cpr(\vec{w_1},\vec{w_2}))_{r_i}) + \frac{\lambda}{2}\sum_j \theta_j^2$

Where θ is a single vector containing all of the entries of all of the model parameters:

(10) $\theta = \text{linearize}([\mathbf{A}^{[\mathbf{1}...\mathbf{D}']}, \mathbf{B}, \vec{c}, \mathbf{K}^{[\mathbf{1}...\mathbf{D}]}, \mathbf{L}, \vec{m}, \mathbf{S}])$

4.3 Gradients for S

The simplest gradient to compute is that of the last parameter to be applied, the matrix $\mathbf{S}$ in the probability function. Here I do so, assuming for simplicity that $N = 1$.

(11) $\frac{\partial E}{\partial S_{j,k}} = \lambda S_{j,k} - \frac{\partial E}{\partial \pi(\vec{\mu}(\vec{y}))_{r_i}} \frac{\partial \pi(\vec{\mu}(\vec{y}))_{r_i}}{\partial S_{j,k}}$

(12) $\frac{\partial E}{\partial \pi(\vec{\mu}(\vec{y}))_{r_i}} = -\frac{1}{\pi(\vec{\mu}(\vec{y}))_{r_i}}$

(13) $\vec{\mu}(\vec{y}) = exp(\mathbf{S} \cdot [1;\vec{y}])$

$$(14)\quad \vec{\pi}(\vec{y})_r = \frac{\vec{\mu}(\vec{y})_r}{\sum_{r'=1}^{R} \mu(\vec{y})_{r'}}$$

if $r \neq j$

$$(15)\quad \frac{\partial \pi(\vec{y})_{r_i}}{\partial S_{j,k}} = \frac{-\vec{\mu}(\vec{y})_r \cdot [1;\vec{y}]_k}{(\sum_{r'=1}^{R} \mu(\vec{y})_{r'})^2}$$

$$(16)\quad \frac{\partial E}{\partial S_{j,k}} = \lambda S_{j,k} - \frac{}{\pi(\vec{y})_{r_i}} \frac{\vec{\mu}(\vec{y})_r \cdot [1;\vec{y}]_k}{(\sum_{r'=1}^{R} \mu(\vec{y})_{r'})^2}$$

if eq

$$(17)\quad \frac{\partial \pi(\vec{y})_{r_i}}{\partial S_{j,k}} = \frac{[1;\vec{y}]_k \cdot \sum_{r'=1}^{R} \mu(\vec{y})_{r'} - \vec{\mu}(\vec{y})_r \cdot [1;\vec{y}]_k}{(\sum_{r'=1}^{R} \mu(\vec{y})_{r'})^2}$$

$$(18)\quad \frac{\partial E}{\partial S_{j,k}} = \lambda S_{j,k} + \frac{1}{\pi(\vec{\mu}(\vec{y}))_{r_i}} \frac{[1;\vec{y}]_k \cdot \sum_{r'=1}^{R} \mu(\vec{y})_{r'} - \vec{\mu}(\vec{y})_r \cdot [1;\vec{y}]_k}{(\sum_{r'=1}^{R} \mu(\vec{y})_{r'})^2}$$

4.4 The comparison function

Since we use $\tanh(x)$ for the sigmoid function f, we have that $\frac{df}{dx} = sech^2(x)$.

References

Marco Baroni and Alessandro Lenci. How we blessed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 1–10. Association for Computational Linguistics, 2011.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 23–32. Association for Computational Linguistics, 2012.

Danqi Chen, Richard Socher, Christopher D Manning, and Andrew Y Ng. Learning new facts from knowledge bases with neural tensor networks and semantic word vectors. *arXiv preprint arXiv:1301.3618*, 2013.

Katrin Erk. Representing words as regions in vector space. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 57–65. Association for Computational Linguistics, 2009.

Christoph Goller and Andreas Kuchler. Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, volume 1, pages 347–352. IEEE, 1996.

Thomas F Icard III. Inclusion and exclusion in natural language. *Studia Logica*, 100(4): 705–725, 2012.

Alessandro Lenci and Giulia Benotto. Identifying hypernyms in distributional semantic spaces. 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*, pages 75–79. Association for Computational Linguistics, 2012.

B. MacCartney and C.D. Manning. An extended model of natural logic. In *Proceedings of the Eighth International Conference on Computational Semantics*, pages 140–156. Association for Computational Linguistics, 2009.

Bill Maccartney. *Natural language inference*. PhD thesis, Citeseer, 2009.

R. Socher, J. Pennington, E.H. Huang, A.Y. Ng, and C.D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics, 2011a.

Richard Socher, Eric H Huang, Jeffrey Pennington, Andrew Y Ng, and Christopher D Manning. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. *Advances in Neural Information Processing Systems*, 24:801–809, 2011b.

Dominic Widdows. *Geometry and meaning*. CSLI Publications, 2004.