Inferential Role Semantics for Natural Language

Peter Blouw (pblouw@uwaterloo.ca)
Chris Eliasmith (celiasmith@uwaterloo.ca)
Centre for Theoretical Neuroscience, University of Waterloo
Waterloo, ON, Canada N2L 3G1

Abstract

Cognitive models have long been used to study linguistic phenomena spanning the domains of phonology, syntax, and semantics. Of these domains, semantics is somewhat unique in that there is little clarity concerning what a model needs to be able to do in order to provide an account of how the meanings of complex linguistic expressions, such as sentences, are understood. To help address this problem, we introduce a tree-structured neural model that is trained to generate further sentences that follow from an input sentence. These further sentences chart out the "inferential role" of the input sentence, which we argue constitutes an important part of its meaning. The model is trained using the Stanford Natural Language Inference (SNLI) dataset, and to evaluate its performance, we report entailment prediction accuracies on a set of test sentences not present in the training data. We also report the results of a simple study that compares human plausibility ratings for both ground-truth and model-generated entailments for a random selection of sentences in this test set. Finally, we examine a number of qualitative features of the model's ability to generalize. Taken together, these analyses indicate that our model is able to accurately account for important inferential relationships amongst linguistic expressions.

Keywords: natural language inference; recursive neural networks; language comprehension; semantics

Introduction

By most accounts, linguistic comprehension is the result of cognitive processes that map between sounds and mental representations of meaning (Christiansen & Chater, 2016; Pickering & Garrod, 2013; Smolensky & Legendre, 2006). An obvious challenge for these accounts is to provide a good theoretical characterization of the relevant representations. Numerous proposals can be found in the literature, but there is no obvious consensus regarding their relative merits.

Arguably, the reason for this lack of consensus is that linguistic comprehension is itself a somewhat vague and ill-defined phenomenon. In the context of efforts to *model* linguistic comprehension, for instance, it is not entirely obvious what a model needs to be able to do in order to provide an account of how people understand complex linguistic expressions such as phrases and sentences.

In this paper, we argue that one thing models of linguistic comprehension need to be able to do is generate predictions about what follows from a given sentence during a conversation. For example, to understand the statement "The dancers parade down the street", one must be able recognize that the dancers are outside, that they are not standing still, that there is likely a surrounding audience, along with various other things. Comprehending a sentence therefore involves drawing inferences that identify the expected consequences of the occurrence of the sentence in the linguistic environment. And since comprehending a sentence involves

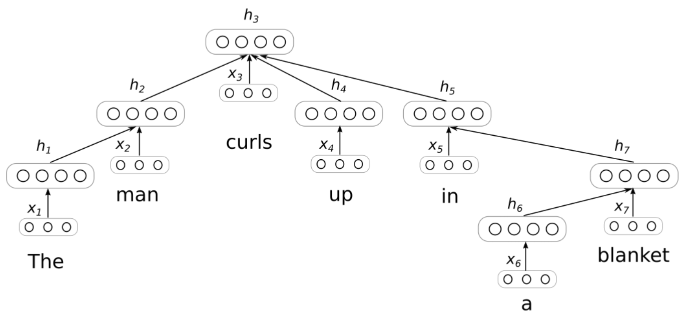

Figure 1: Sentence encoding with a dependency tree recursive neural network (DT-RNN). A dependency parser is used to produce the computational graph for a neural network, which is then used to produce a distributed representation of sentence by merging distributed representations of individual words. Figure adapted from Socher et al. (2014).

comprehending its *meaning*, it follows that meaning of an expression is at least partly determined by the inferences it licenses (Brandom, 1994)

To motivate this inferential approach to semantics, we introduce a neural network model that learns to generate sentences that are the inferential consequences of its inputs. The model functions by first encoding a sentence into a distributed representation, and then decoding this representation to produce a new sentence. The encoding procedure involves dynamically generating a tree-structured network layout of the sort depicted in Figure 1. Once a sentence encoding is produced using this network, it is fed through an "inverse" tree-structured network to produce a predicted sentence. Interestingly, different inverse or decoding networks can be used to generate different sentences from a single encoding. To train the model parameters (i.e. the network weights shared across different tree structures) we use the Stanford Natural Language Inference dataset (Bowman et al., 2015).

In what follows, we first describe the model and then empirically evaluate its ability to produce plausible entailments for sentences unseen in the training data. We present experimentally produced plausibility ratings for a random collection of generated sentences, and from these ratings conclude that the model captures something important about the inferential roles of ordinary linguistic expressions. We further contend that the model motivates the view that understanding a linguistic expression is *not* (as is typically thought) a matter of mapping it onto a representation that somehow constitutes its meaning. Rather, understanding a linguistic expression is a matter of inferring the expected consequences of its occurrence in the linguistic environment. The reason for drawing this conclusion is that the expected consequences of a sentence cannot be "read off" of any single representation in the model. Instead, these consequences are derived from the global behavior of the model and the processes that it implements.

Tree-Structured Neural Networks

To build our model, we take advantage of recently developed techniques for using neural networks to define composition functions that merge distributed representations of words into distributed representations of phrases and sentences (Socher et al., 2012, 2014). The core idea behind these techniques is to produce a parse tree for a sentence, and then transform the tree into a neural network by replacing its edges with weights and its nodes with layers of artificial neurons. Activation is then propagated up the tree by providing input to layers that correspond to certain nodes, as shown in Figure 1. The input at each node is typically a distributed representation or "embedding" corresponding to a single word (see Mikolov et al., 2013).

This general method can be applied using arbitrary tree structures, and we adopt a dependency-based syntax in the experiments described below. There are three reasons for this choice (Socher et al., 2014). First, the assignment of different network weights to different dependency relations allows for the creation of networks that are more sensitive to syntactic information. Second, the semantic role of an individual word can often be read off of the dependency relation it bears to a head word, which allows for the creation of networks that are also sensitive to semantic information. Finally, dependency trees are less sensitive to arbitrary differences in word order, which helps to ensure that simple variations of a sentence get mapped to similar distributed representations. The model we adapt - the dependency tree recursive neural network (DT-RNN) - is introduced in Socher et al. (2014)

Some formal details concerning the behavior of DT-RNNs are helpful at this point. First, an input sentence s is converted into a list of pairs, such that $s = [(w_1, x_1), (w_2, x_2), ...(w_n, x_n)]$, where w is a word and x is the corresponding word embedding. Next, a dependency parser is used to produce a tree that orders the words in the sentence in terms of parent-child relations. Each node in this tree is then assigned an embedding in a two-step manner. First, all of the leaf nodes in the tree (i.e. nodes that do not depend on other nodes) are assigned embeddings by applying a simple transformation to their underlying word embeddings:

$$h_i = f(W_v x_i + b) \tag{1}$$

where h_i is the embedding for some leaf node i in the tree, x_i is the embedding for the word corresponding to this node, W_v is a matrix that transforms word representations, b is a bias term, and f is an element-wise nonlinearity. Second, embeddings are recursively assigned to all of the non-leaf nodes by composing the embeddings of their children as follows:

$$h_i = f(W_v x_i + \sum_{j \in C(i)} W_{R(i,j)} \cdot h_j + b) \tag{2}$$

where h_i is again the embedding for some node i in the tree, x_i is the embedding for the word corresponding to this node, j is an index that ranges over the children, $C(i)$, of the node i, and $W_{R(i,j)}$ is a matrix associated with the specific dependency relation between node i and its j^{th} child. h_j is the embedding corresponding to this child. So, in the example tree in Figure 1, the embeddings for nodes 1, 4, and 6 would be computed first, since these nodes have no children. Then, embeddings will be computed for any nodes whose children now all have assigned embeddings (in this case, nodes 2 and 7). And so on, until an embedding is computed for every node.

Model training is done via backpropogation and requires that a cost function be defined for the sentence embeddings produced at the root of each tree. The free parameters are the weights W_v and $W_{r \in R}$, along with the bias term b. Word embeddings can also be fine-tuned over the course of training.

Generating Entailments

Choosing an appropriate cost function for a recursive neural network can be difficult, since it is not always clear what makes for a "good" sentence embedding. It is accordingly common to see these networks applied to narrow classification tasks such as the prediction of sentiment ratings (e.g. Socher et al., 2012). Our goal is define an optimization objective that accounts for the principle that understanding a linguistic expression involves drawing inferences about what follows from it.

To accomplish this goal, we define a model composed of two DT-RNNs, one that encodes an input sentence into a distributed representation, and another that decodes this representation into a new sentence that is entailed by the input sentence. This model is inspired by Iyyer et al.'s (2014) work using DT-RNNs analogously to autoencoders, but introduces a decoding procedure that computes an appropriate response to the input sentence, rather than merely reconstructing it. Other related work is described in (Kolesnyk et al., 2016).

The model is trained on pairs of sentences standing in entailment relations. A dependency parser[1] is again used to produce a tree-structured network for each sentence, but the network associated with the second sentence is run in reverse, as shown in Figure 2. A word prediction is generated at each node in this second tree using a softmax classifier, which allows us to define a cross-entropy loss function over nodes and trees as follows:

$$J(\theta) = -\sum_i \sum_j t_j^{(i)} \log p(c_j^{(i)} | s_i) \tag{3}$$

[1]We use the SpaCy python library, available at https://spacy.io

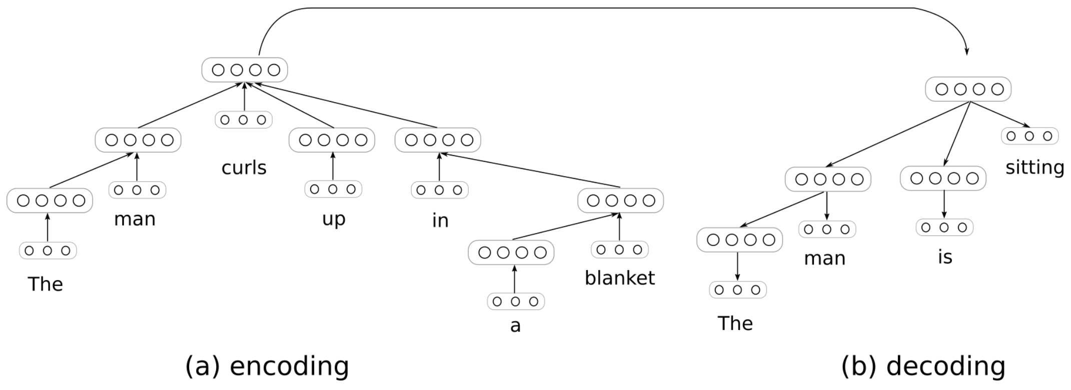

Figure 2: Generating entailments with paired encoder and decoder DT-RNNs. The decoder network computes a probability distribution over words at each node, conditioned on the sentence representation produced by the encoder. The parameters of both the encoder and decoder are trained via backpropogation through structure using error derivatives supplied at each node in the decoding tree. The encoder and decoder trees are dynamically generated for each pair of sentences in the training data.

where $t_j^{(i)}$ is the target probability (i.e. 1) for the correct word at the j^{th} node in the i^{th} training example, $p(c_j^{(i)}|s_i)$ is the computed probability for this word given the input sentence s_i, and θ is the set of combined parameters for the encoder and decoder DT-RNNs.

We train the model via stochastic gradient descent by backpropogating through both the decoder and encoder tree for each training example. The result of training is a set of weights associated with dependencies for both encoding and decoding, a set of weights for predicting a distribution over words from a node embedding for each dependency, a set of biases (we allow dependency-specific biases), and the input transformation matrix W_v. When the trained model is used to perform inference using a novel input sentence, the encoder DT-RNN is assembled into a tree using the learned encoding weights. The decoder DT-RNN is then also assembled into a tree using the learned decoding weights, and activation is propagated through the encoder and into the decoder to produce a probability distribution over words at each tree node. The words with the highest probability at each node are then used to construct the predicted entailment for the input sentence. The tree structure for the decoder can either be selected randomly or stipulated ahead of time.

Experiments

In the remainder of the paper, we describe a number of basic experiments that illustrate how this general modeling framework can be used to illuminate the phenomenon of language comprehension. We first perform a basic evaluation of how well the decoder model is able to generate entailments by measuring the percentage of correct word predictions over all decoding tree nodes in both the training set and an unseen test set. We then present the results of an experiment designed to evaluate the quality of the entailments generated by our model. Next, following Kolesnyk et al. (2016), we iterate the encoding-decoding procedure to generate chains of entailments from a given input sentence that delineate simple inferential roles. Finally, we analyze the effect of substituting individual words in an input sentence. The goal of this analysis is to evaluate the extent to which the model is able to learn indirect inferential roles for words and appropriately generalize to a wide range of novel sentences that can be substitutionally derived from a single familiar sentence.

Training Data

To train encoder and decoder networks, we use a subset of the Stanford Natural Language Inference dataset introduced in Bowman et al. (2015). This dataset consists of approximately 570,000 sentence pairs with labeled inferential relationships. Specifically, the first sentence in each pair can either entail, contradict, or be neutral with respect to the second sentence, and since our interest is generating entailments, we restrict our attention to pairs labeled with the entailment relation.

To reduce the amount of noise and complexity in the dataset, we also perform some simple pre-processing steps. First, we screen for misspelled words,[2] and eliminate all sentence pairs containing a misspelling. Second, we eliminate all sentence pairs containing a sentence longer than 15 words in order to avoid fitting model parameters to a small number of very long sentences that produce highly complex dependency trees. After preprocessing, the data consists of 106,288-pair training set, a 1701-pair development set, and 1666 pair test set. We train on the training set and use the development set for tuning hyperparameters such as the learning rate and the number of training epochs. The vocabulary used during training and testing consists of 22,555 words.

Quantitative Evaluations

To evaluate the ability of the model to generate plausible entailments, we first measure the proportion of correct word-level predictions during decoding in both the training set and

[2]We use the PyEnchant python library, available at http://pythonhosted.org/pyenchant/.

Table 1: Examples of Entailments Generated From Novel Test Sentences.

Sentence	A boy and girl child swing together on a swing set.	A young blond boy is eating cake with a spoon.
Entailment	Two kids swing on a swing.	A boy is eating a cake.
Sentence	A young man sleeping next to a dog	A surfer is performing a jump stunt in the ocean.
Entailment	A man is near a dog.	A surfer and a surfboard is outside.

Table 2: Word-Level Accuracy for Entailment Generation

Model	Training Set (%)	Test Set (%)
Chance	6.0	5.9
DT-RNN	66.7	61.8

Table 3: Plausibility Ratings for Inferential Relations.

Source	Status	Mean Likert Rating (1-5)
Human	Entailment	4.05 ± 0.09
Model	Entailment	3.53 ± 0.12
Human	Contradiction	2.05 ± 0.12

* Margins are bootstrapped 95% confidence intervals.

the test set. We provide the tree structure of each entailed sentence during decoding, so inference involves propagating activities through paired trees of the sort depicted in Figure 2 to generate a set of word predictions. Some example entailments produced from sentences drawn from the test set are listed in Table 1. The decoding tree used to produce each entailment is chosen randomly in these examples.

The word vectors that provide input to the encoder are initialized using 300-dimensional Word2Vec embeddings (Mikolov et al., 2013), while biases are initialized as the zero vector. Each set of weights associated with a syntactic dependency is initialized as a 300×300 identity matrix with mean-zero Gaussian noise for both the encoder and decoder. The word transformation matrix, W_v, is initialized in the same way. During learning, all of these matrices are updated using stochastic gradient descent, along with the Word2Vec embeddings and the biases. We perform approximately 5 epochs of training using an initial learning rate of 6×10^{-4}, and we progressively anneal this rate over the course of training.

To collect accuracy measures, we simply tally the proportion of nodes in the decoding trees for which the predicted word is the same as the actual word given in the relevant test set; the decoding tree is determined by a parse of the correct entailment in every case. We compare against a baseline accuracy of chance. As shown in Table 2, The DT-RNN model performs considerably better. It is worth noting that generated sentences containing words not present in the correct entailment may still be appropriate, given that no entailment is *uniquely* correct. It is also worth noting that prior work involving SNLI has almost uniformly focused on the problem of classifying sentence pairs. Given that our interest is in generation rather than classification, we cannot easily draw comparisons to earlier work, and therefore use novel methods of evaluation.

Empirical Evaluations

Next, we conduct a simple study in which human subjects are asked to evaluate the plausibility of model-generated sentences. During the study, participants are shown a series of sentences introduced as true captions for unseen images.[3] For each caption, the participants are shown an alternate caption and asked to evaluate the likelihood that it is also true of the corresponding image. Evaluations are recorded using a five point Likert scale that ranges from "Extremely Unlikely" (1) to "Extremely Likely" (5). The original caption in each case is the first sentence in a pair randomly chosen from the SNLI test set, while the alternate captions are either (a) model-generated entailments, (b) human generated entailments drawn from the test set, or (c) human generated contradictions also drawn from the test set. This between-subjects experimental design is similar to the method used by Bowman et al. (2015) to validate human-generated sentence pairs during the creation of SNLI. The main difference is that we evaluate model-generated sentences in addition to human-generated sentences.

Seventy-five participants from the United States were recruited through Amazon's Mechanical Turk and split evenly into the three conditions. The main captions were identical across conditions, and each participant was asked to rate 20 caption pairs.[4] Participants were paid $1.00 for their time. Two of the seventy-five participants failed to complete the study and did not have their responses included in the results. Repeat participation was blocked by screening Mechanical Turk worker IDs.

The Likert ratings collected during the study are assessments of the plausibility of the inferential transition from one sentence (the main caption) to another (the alternate caption).

[3]Note that all of the sentence pairs in SNLI were generated by providing subjects with a caption for an unseen image and asking them to produce a further caption that is either true, false, or maybe true of the image. So all of the sentences in SNLI can be described as image captions. The point of using this caption-based strategy in the construction of the dataset is to eliminate co-reference ambiguities that make it difficult to determine the appropriate inferential relationship between two sentences. See Bowman et al. (2015) for more details.

[4]Two of the main captions had no associated contradictions in SNLI, so subjects in the contradiction condition only rated 18 captions.

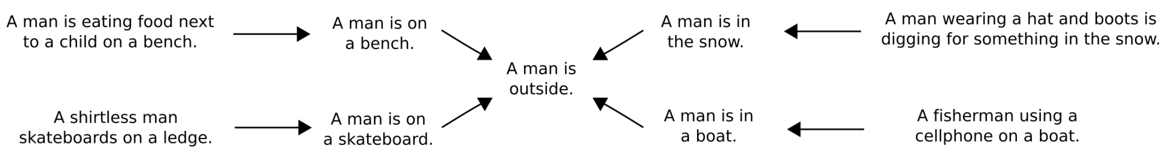
A man is eating food next to a child on a bench.
A man is on a bench.
A shirtless man skateboards on a ledge.
A man is on a skateboard.
A man is outside.
A man is in the snow.
A man wearing a hat and boots is digging for something in the snow.
A man is in a boat.
A fisherman using a cellphone on a boat.

Figure 3: A model-generated inferential network around the sentence "A man is outside". Each inferential transition is the result of generating a predicted entailment after encoding the sentence at the beginning of each arrow. The entire network is generated starting with only the four outermost sentences, which are drawn from the SNLI test set.

The transitions involving sentence pairs drawn directly from SNLI offer a kind of gold standard for both good and bad transitions. The results shown in Table 2 indicate that model-generated transitions are seen to be almost as plausible as the gold-standard transitions drawn from SNLI. We take this to be preliminary evidence that model is able to capture certain tacit inferential relationships between natural language expressions.

Qualitative Extensions

In order to further analyze the model's behavior, we examine a number of qualitative features of the inferential relationships it is able to learn. First, we examine iterative applications of the model to its own predictions, following similar work by Kolesnyk, Rocktäschel, and Reidel (2016) that makes use of a sequential LSTM. Next, we examine the model's ability to disciminate the inferential significance of lexical items by performing simple word-by-word substitutions in an input sentence. The point of these analyses is to demonstrate that models of the general class we are proposing are useful tools for both formalizing and learning the inferential roles of a wide variety of linguistic expressions.

Iterative Inferences

Once an input sentence has been passed through the model to generate an entailment, it is possible to use this entailment as a new input to the model. Repeated applications of the model accordingly make it possible to chart out an "inferential network" around a particular starting sentence. Figure 3 offers a simple model-generated example of an inferential network in which numerous sentences describing men doing things outdoors are eventually mapped to the sentence "A man is outside".

In general, predicted entailments that are shorter than an input sentence tend to be more abstract and general, while predicted entailments that are longer than an input sentence tend to introduce plausible elaborations (Kolesnyk et al., 2016). For instance, the sentence "A bird is in a pond." can be used to generate the sentence "A little bird is outside in a small pond." by using a decoding tree with nodes for two additional adjectives and an additional adverb.

Substitutional Analysis

If individual words in an input sentence are replaced, it becomes possible to identify the impact of particular words on the inferences that are licensed by a particular sentence. In Figure 4, for instance, the replacement of a subject noun or the main verb can be seen to have significant effects on the kinds of entailments that are generated. The model is impressively sensitive to sophisticated linguistic cues concerning agreement. For instance, the model correctly infers that "boy" should be paired with the male possessive "his", while "girl" should be paired with the female possessive "her". It is worth emphasizing that all of the sentences that result from substitution are completely novel from the model's perspective. The fact that the model is able to generate reasonable entailments for many of these sentences suggests that it is able generalize beyond the training data quite successfully.

A further application of substitutional analysis involves examining a model's ability to learn about theoretically interesting constructions involving negations, quantifiers, and numerals. For instance, the model exhibits a rudimentary ability to handle numerals appropriately, as is shown by the inference from "A boy and a girl..." to "Two kids..." Negations are a bit more troublesome: the model correctly infers "not outside" from "in a car", but incorrectly infers "not indoors" from "in a store". Quantifiers, finally, are an open question: the model correctly infers "The women" from "Many women", but it is not clear that this is the result of learning a relation between "Many" and the plural forms of nouns. Examining specific linguistic constructions in this substitutional manner is a promising avenue for future research.

Discussion

Overall, the point of this work is to motivate an approach to semantics based on inferential relationships amongst linguistic expressions (Brandom, 1994). Our use of the encoder-decoder DT-RNN model is designed to illustrate how generalized inferential roles can be learned for arbitrary linguistic expressions from examples of how sentences are distributed as tacit "premises" and "conclusions" in a space of inferences. It is accordingly possible to characterize this work as an extension to the well-known distributional approach to semantics (Turney & Pantel, 2010), wherein we replace the generic notion of a linguistic context with the more fine-grained notion of an inferential context.

As with most natural language generation systems, many of the sentences produced by our model are defective in some way. As can be seen in the examples in Table 1, our generated entailments are almost always thematically appropriate,

A boy in a beige shirt is sleeping in a car.	⟶	A boy is in his car.	Some men in red shirts are waiting in a store.	⟶	The men are in a store.
A girl in a beige shirt is sleeping in a car.	⟶	A girl is in her car.	Many women in red shirts are waiting in a store.	⟶	The women are in a store.
A man in a beige shirt is sleeping in a car.	⟶	A man sleeping in his car.	A boy and a girl are waiting in a store.	⟶	Two kids are indoors
A woman in a beige shirt is sleeping in a car.	⟶	A woman is in her car.	A boy and a girl are waiting in a playground.	⟶	Two kids are outside
A man in a beige shirt is driving in a car.	⟶	A man is driving a car.	A boy in a red shirt is sleeping in a car.	⟶	A boy is not outside.
A person in a beige shirt is selling a car.	⟶	A person is selling a car.	A boy in a red shirt is waiting in a store.	⟶	A boy is not indoors.

Figure 4: Substitutional analysis using the sentence "A boy in a beige shirt is sleeping in a car". The model is able to predict appropriate entailments for a range of sentences that are similar to this initial sentence shown at the top left. The fact that these substitutionally-derived sentences are not present in SNLI dataset indicates that our model is able to generalize by interpolating between the example inferential transitions found in the training data.

but sometimes contain agreement errors or misplaced words that render the entailment as a whole ill-formed. And, not infrequently, the model produces entailments that are more or less incomprehensible. There are two ways to address these problems. The first involves the use of increased amounts of training data to provide the model with a more points in the "space of inferences" to interpolate between. The second involves the use of more sophisticated network architectures that help the model to learn to more selectively make use of only the input information that is most relevant to generating a good entailment. LSTM network architectures, such as the Tree LSTM (Tai et al., 2015), are likely to provide improvements on this second front.

Finally, an important limitation of our work is that we do not consider the relationship between linguistic expressions and the non-linguistic world. A natural way to account for this relationship is to suppose that a sentence's occurrence in the linguistic environment licenses certain expectations about what can be seen, heard, or otherwise perceived. To return to our initial example, if one understands the statement "The dancers parade down the street", one will expect to see and hear dancers upon going to the relevant street. We accordingly suggest that if an individual can adequately infer all that follows from a given linguistic expression, both linguistically and non-linguistically, then there is nothing further they need to be able to do to count as *understanding* what the expression means. The main consequence of this view is that inference should be at the core of any theory of semantic cognition.

Acknowledgments

This work was supported by an Ontario Graduate Scholarship and the Canada Research Chairs program.

Code

All of the simulations described in this paper were implemented using a neural network library written by the first author, available online at https://github.com/pblouw/pysem.

References

Bowman, S., Angeli, G., Potts, c., & Manning, C. (2015). A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 conference on empirical methods in natural language processing.* Association for Computational Linguistics.

Brandom, R. (1994). *Making it explicit: Reasoning, representing, and discursive commitment*. Harvard University Press.

Christiansen, M., & Chater, N. (2016). The now-or-never bottleneck: A fundamental constraint on language. *Behavioral and Brain Sciences*, 1-72.

Iyyer, M., Boyd-Graber, J., & Daume III, H. (2014). Generating sentences from semantic vector space representations. In *Nips workshop on learning semantics.*

Kolesnyk, V., Rocktäschel, T., & Reidel, S. (2016). Generating natural language inference chains. *arXiv preprint arXiv:1606.01404.*.

Mikolov, T., Sustkever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems 28.*

Pickering, M., & Garrod, S. (2013). An integrated theory of language production and comprehension. *Behavioral and Brain Sciences*, *36*, 329-392.

Smolensky, P., & Legendre, G. (2006). *The harmonic mind: From neural computation to optimality-theoretic grammar* (Vol. 1). MIT Press.

Socher, R., Huval, B., Manning, C., & Ng, A. (2012). Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning* (p. 1201-1211). Association for Computational Linguistics.

Socher, R., Karpathy, A., Le, Q., Manning, C., & Ng, A. (2014). Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association of Computational Linguistics*, *2*, 207-218.

Tai, K., Socher, R., & Manning, C. (2015). Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd annual meeting of the association for computational linguistics* (p. 1556-1566). Association for Computational Linguistics.

Turney, P., & Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, *37*, 141-188.