Definition Learning

Colin McDonnell

May 13, 2016

1 Introduction

Deep learning has enabled unprecedented improvements across the field of NLP in recent years. It has improved results in speech synthesis, speech-to-text systems, and machine translation. This is all possible because of the idea of continuous-valued distributed representations of words, also called word embeddings or vectors. Associating each word with some vectors of a thousand scalars has certainly led to some cool tricks. We can consider distance in this thousand-dimensional space as a approximation of "word similarity". We've seen some interesting linear properties emerge, like the canonical vec("king") - vec("man") + vec("woman") = vec("queen"). Less obvious compositional capabilities have also been observed, for instance vec("Russia") + vec("river") = vec("Volga").

2 Problem

Deep learning pioneer Yoshua Bengio recently said "I think we'll see much more in the upcoming years, moving more and more towards the fundamental challenge of natural language UNDERSTANDING" (emphasis his, from Quora AMA). Bengio doesn't claim, as some have, that the embedding models truly understand the natural language they process. This claim makes sense when you look at the architectures and training rules used to learn these vectors.

The fundamental task being performed by these networks is prediction: based on some words in the text, predict nearby words. This comes in two flavors, depicted well in Figure 1.

The bag-of-words model looks at surrounding words and predicts the middle word. The skip-gram model predicts surrounding words based on a given word [4]. Interestingly, the skip-gram model is the state of the art, which gives us a really good indication of the sorts of features these systems are learning. The model is given a single word, often without context, and told to predict surrounding words. What is really learned by a network that has maximized this objective function? A table of co-occurrence statistics, just like those used by old-school n-gram models.

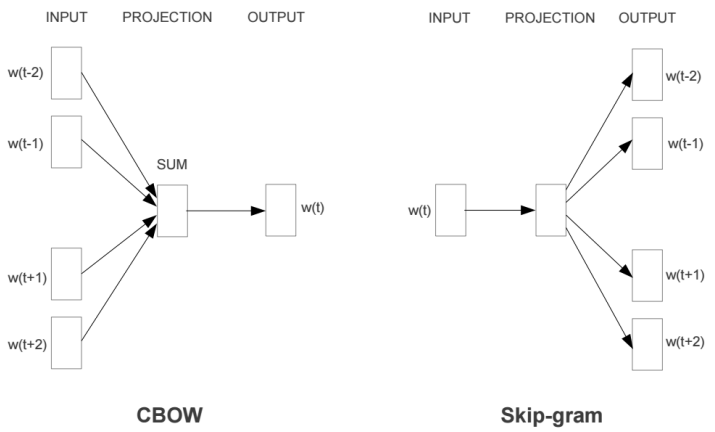

Figure 1: Two architectures for learning word embeddings. a) The left represents the continuous bag-of-words model. b) The right represents the skip-gram model.

Deep learning systems "cheat". They can converge to a system that perform well on semantic relatedness tests, sentiment analysis, and sentence classification tasks by doing clever things with co-occurrences statistics. However that is qualitatively different from extracting information from text and putting it into a world model that can be operated on logically and compositionally. No system understands what a word or sentence really means. At risk of sounding like John Searle, we define "understanding" as the ability of a system to independently construct a verbal definition for a word or concept from its internal representation.

The challenge is thus determining what data, architecture, and training method would force a neural network to learn such complex, semi-symbolic representations. This is hard. As we move towards more human scale capabilities, datasets grow scarcer. For instance, Bengio's lab is working on a system that parses the Ubuntu user manual and can answer questions on its contents. Few humans have the time or inclination to do perform tasks like that, so labeled training data necessarily gets more expensive. So to summarize, the unsupervised prediction rules are too weak to force neural networks to develop rich representations, and a theoretical "labeled dataset" that captures the essence of human-level reasoning and text interpretation doesn't exist. So, how do we move forward?

We can start by looking at how humans learn language. As children, we probably encounter a new word, ask what it means, are told the definition of a thing. After seeing it used in a sentence a handful of times, our internal representation is more or less mature. The key point here is that we are told a definition of a thing, and that definition gives us some priors over which we can make predictions regarding the proper usage of the term.

Another interesting point here is children rarely read these definitions from a

dictionary; they are told by a teacher or parent who is familiar with what the child already knows and how to describe the new concept in terms of previously taught information. This is an important point we'll revisit later.

So the goal of this project is to teach a neural network to learn explicitly from word definitions instead of implicitly from large corpora of words. More broadly, we want to build a system that understands the meanings of individual words and can infer, understand, and describe the compositional structure of verbal concepts.

3 Approach

The approach is simple. We teach a network to construct a representation of the word based only on the words in its definition. This forces the neural network to learn the true definition of a word, because that is the only information available. Because the definitions vary in size, we need some way to accept variable length input. We examine the approach in more detail below.

3.1 Representations

We left the representations of both the definition and target vague above, however we intend to use pre-trained vectors from existing word embedding systems for both. It is a bit counter-intuitive that we are using existing embeddings to train this network, considering that we just discussed at length the inadequacy of the models that generated them. The rationale for this is a bit subtle.

Though the methodology used to generate these word vectors does not result in a system that can naturally manipulate a rich human-level representation, the information to generate such a representation is contained within the word vectors. Intuitively speaking, the information required to construct a human-level representation of a word is there, but we don't know how to access it. There is some supporting evidence for this. In the paper Skip-Thought Vectors [3], Kiros et al learned a set of embeddings (called skip-thought vectors) for a small vocabulary using a methodology that was effective yet computationally expensive. The resulting vectors performed better than the word2vec vectors generated my Mikolov et al, though the vocabulary was an order of magnitude smaller. Kiros trained a linear network to map from word2vec vectors to skip-thought vectors, using only the words they had in common. Then, she used this mapping network to generate skip-thought vectors for every word in the word2vec vocabulary. These generated vectors were nearly as good as the "natural" skip-thought vectors, and performed better than the word2vec vectors from which they were generated on many syntactic and semantic benchmarks.

Extrapolating, these word embeddings contain a huge amount of information about the words they represent. And they should, because they've been trained on billions of words of text, far more than any human could ever read. Remember our definition of "understanding" as "ability of a system to independently construct a verbal definition for a word from its internal representation". In

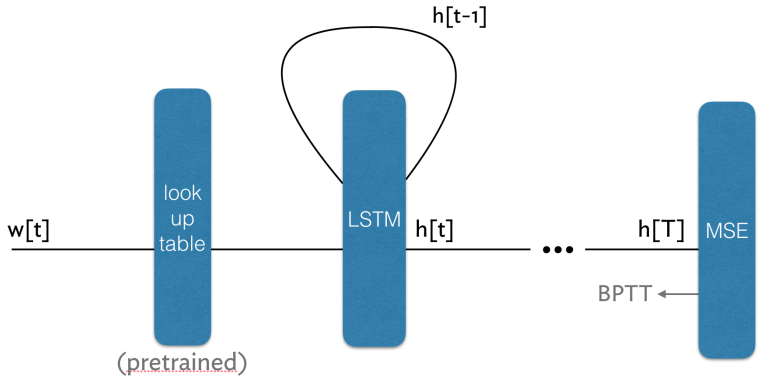

Figure 2: Depiction of the network used in this work.

current systems, even though the internal representation contains the necessary information to construct a definition, no one knows how to ask for it. More generally, it remains difficult to find ways to compose that information into a meaningful representation. In any case, we don't see any problem in using these word vectors as a "ground truth" value for these words, because they probably contain most or all of the relevant information about a given word.

3.2 Handling variable length definitions

There are many ways to handle variable length definitions. We settled on a standard LSTM network with sequence-to-one training. Under this training regime, a sequence of inputs are fed into the network. After the entire sequence is read in, the current activation of the hidden layer is interpreted as the output of the model. Training gradients are generated based on the different between this output and the desired output, and those gradients are passed back through the network according the the backpropagation through time (BPTT) algorithm. It's important that the number of steps through which the gradients propagate is greater than or equal to the length of the longest sequence input.

The input layer is a lookup table containing pre-trained word vectors. This table can be updated as more expressive word vectors emerge in the literature. The middle layer is a LSTM hidden layer with 50 nodes. The output is generated after the entire sequence of length T has been fed into the network. The error is computed using a mean-squared error loss criterion. This is represented pictorially in Figure 2.

4 Results

The research is ongoing. Check back later.

We intend to use train this network on half of Webster's dictionary with mini-batch stochastic gradient descent. This should give us an encoder network that can accept as input a dictionary and output a reasonable word vector. We can use the other half of the dictionary for validation and testing. We can evaluate the generated vectors on standard benchmark tasks for semantic and syntactic relations. We can also compute the perplexity of the language model as an indication of representation quality.

5 Future extensions

There is a huge number of ways to extend this research (besides actually getting results).

This network can be paired with a "definition detector" network that can detect when a given sentence in a corpus is defining a term or commenting on it explicitly, as opposed to simply using it in its proper context. This network could then give a disproportionate weight to the updates performed on that particular sentence. These sorts of innate biases have proven to be extremely useful in achieving high performance in other tasks such as vision [5].
We also hope to incorporate another idea from Bengio: curriculum learning [1]. In his words, "Humans and animals learn much better when the examples are not randomly presented but organized in a meaningful order which illustrates gradually more concepts, and gradually more complex ones". As we mentioned before, a parent or teacher will define a word in terms of other word the child already knows. In this sense, the adult takes the randomly presented linguistic stimulus available to the child and curricularizes it. This can be performed in an automated way by doing a topological sort of the dictionary before training. This means that, to the extent possible, we train the network to learn words that are defined in terms of other words it has already seen, just like a child. This will hopefully enhance learning.

Of course, all words can't be defined entirely in terms of other words to avoid circular definitions. There must be some set of irreducible terms whose meaning is inferred from its use in context. Among these might include articles and prepositions. We can then train networks that have been initialized with different initial linguistic knowledge.

There are many other ways to extend this research. One could experiment with different network architectures, for instance a recurrent convolutional networks with temporal pooling to collapse an arbitrary-length definition to a constant length vector as implemented at the character level in [2]. It would be interesting to train new word vectors from scratch instead of using a pre-trained lookup table. It would also be interesting to train a network that performs the opposite training task; that is, based on the embedding of a word, it outputs some representation of its definition. This is somewhat analogous to the skip-gram model described above. One could also look into recursive neural networks, alternative loss functions, other datasets, and other languages.

References

[1] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48, New York, NY, USA, 2009. ACM.

[2] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush. Character-Aware Neural Language Models. *ArXiv e-prints*, August 2015.

[3] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. *CoRR*, abs/1506.06726, 2015.

[4] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[5] Shimon Ullman, Daniel Harari, and Nimrod Dorfman. From simple innate biases to complex visual concepts. *Proceedings of the National Academy of Sciences*, 109(44):18215–18220, 2012.