Fine-grained analysis of sentence embeddings using auxiliary prediction tasks

Yossi Adi[1,3]*, Einat Kermany*[3]*, Yonatan Belinkov*[2]*, Ofer Lavi*[3]*, Yoav Goldberg*[1]

[1]Bar-Ilan University, Ramat-Gan, Israel
[2]Massachusetts Institute of Technology, Cambridge, USA
[3]IBM Haifa Research Lab, Haifa, Israel

adiyoss@cs.biu.ac.il

Abstract

While *sentence embeddings* or *sentence representations* play a central role in recent deep learning approaches to NLP, little is known about the information that is captured by different sentence embedding learning mechanisms. We propose a methodology facilitating fine-grained measurement of some of the information encoded in sentence embeddings, as well as performing fine-grained comparison of different sentence embedding methods.

In sentence embeddings, sentences, which are variable-length sequences of discrete symbols, are encoded into fixed length continuous vectors that are then used for further prediction tasks. A simple and common approach is producing word-level vectors using, e.g., word2vec [1, 2], and summing or averaging the vectors of the words participating in the sentence. This continuous-bag-of-words (CBOW) approach disregards the word order in the sentence. Another approach is the *encoder-decoder* architecture, producing models also known as *sequence-to-sequence* models [3, 4, 5]. In this architecture, an *encoder* network (e.g. an LSTM) is used to produce a vector representation of the sentence, which is then fed as input into a *decoder* network that uses it to perform some prediction task (e.g. recreate the sentence, or produce a translation of it). The encoder and decoder networks are trained jointly in order to perform the final task.

Some systems (for example in machine translation) train the system end-to-end, and use the trained system for prediction [5]. Such systems do not generally care about the encoded vectors, which are used merely as intermediate values. However, another common case is to train an encoder-decoder network and then throw away the decoder and use the trained encoder as a general mechanism for obtaining sentence representations. For example, an encoder-decoder network can be trained as an auto-encoder, where the encoder creates a vector representation, and the decoder attempts to recreate the original sentence [6]. Similarly, in [7] the authors train a network to encode a sentence such that the decoder can recreate its neighboring sentences in the text. Such networks do not require specially labeled data, and can be trained on large amounts of unannotated text. As the decoder needs information about the sentence in order to perform well, it is clear that the encoded vectors capture a non-trivial amount of information about the sentence, making the encoder appealing to use as a general purpose, stand-alone sentence encoding mechanism. The sentence encodings can then be used as input for other prediction tasks for which less training data is available [8]. In this work we focus on these "general purpose" sentence encodings.

The resulting sentence representations are opaque, and there is currently no good way of inspecting different representations short of using them as input for different high-level semantic tasks (e.g. sentiment, entailment, document retrieval, question answering, sentence similarity, etc.) and measuring how well they perform on these tasks. This is the approach taken by [6], [9] and [7]. This method of comparing sentence embeddings leaves a lot to be desired: the comparison is at a very coarse-grained level, does not tell us much about the kind of information that is encoded in the representation, and does not help us form generalizable conclusions.

Our Contribution We take a first step towards opening the black box of vector embeddings for sentences. We propose a methodology that allows to investigate sentence embeddings on a much finer-grained level, and demonstrate its use by analyzing and investigating different sentence representations. We investigate sentence representation methods that are based on LSTM auto-encoders, the skip-thought embeddings of [7], and the simple CBOW representation produced by averaging word2vec word embeddings. For the LSTM and CBOW embeddings, we compare different dimensions, exploring the effect of the dimensionality on the resulting representation.

The main idea of our method is to focus on isolated aspects of sentence structure, and design experiments to measure to what extent each aspect is captured in a given representation. In each experiment, we formulate a pre-

diction task. Given a sentence representation method, we create training data and train a classifier to predict a specific sentence property (e.g. their length) based on their vector representations. We then measure how well we can train a model to perform the task. The basic premise is that if we cannot train a classifier to predict some property of a sentence based on its vector representation, then this property is not encoded in the representation (or rather, not encoded in a useful way, considering how the representation is likely to be used). The tasks are designed such that it is easy to produce an unlimited amounts of training and test data from raw, unannotated text.

In this work, we focus on what are arguably the three most basic characteristics of a sequence: its length, the items within it, and their order. While these are rather low-level properties, our analysis already leads to interesting, actionable insights, such as:

- LSTM auto-encoders are very effective at encoding word order information, and less so at encoding word content.
- Increasing the dimensionality of the LSTM encoder does not significantly improve its ability to encode length, but does increase its ability to encode word content and word order information.
- 500 dimensional embeddings are already quite effective for encoding word order, with little gains beyond that.
- Word content accuracy peaks at 750 dimensions and drops at 1000, suggesting that larger is not always better.
- The trained LSTM encoder (when trained with an auto-encoder objective) does not rely on ordering patterns in the training sentences when encoding them. The skip-thought encoder does rely on such patterns.
- The encoder-decoder's ability to recreate sentences (BLEU) is not entirely indicative of the quality of the encoder at representing aspects such as word identity and order.

Generalizing the approach to higher-level semantic and syntactic properties holds great potential, which we hope will be explored in future work, by us or by others.

1. References

[1] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[2] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[3] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[5] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[6] J. Li, M.-T. Luong, and D. Jurafsky, "A hierarchical neural autoencoder for paragraphs and documents," *arXiv preprint arXiv:1506.01057*, 2015.

[7] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, "Skip-thought vectors," in *Advances in Neural Information Processing Systems*, 2015, pp. 3276–3284.

[8] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning," in *Advances in Neural Information Processing Systems*, 2015, pp. 3061–3069.

[9] F. Hill, K. Cho, and A. Korhonen, "Learning distributed representations of sentences from unlabelled data," *arXiv preprint arXiv:1602.03483*, 2016.