Deep contextualized word representations

1 Citation

Peters, Matthew E., et al. ”Deep contextualized word representations.” arXiv preprint arXiv:1802.05365 (2018).

`https://arxiv.org/pdf/1802.05365.pdf`

2 Abstract

We train a deep bidirectional language model (biLM) and use the internal hidden state vectors as word embeddings. These embeddings get state of the art on six natural language processing (NLP) tasks.

3 Introduction

We want word vectors that capture syntax, semantics, and polysemy (i.e. a word can mean different things depending on the context, like how “play” can refer to playing a game or to a theatrical performance). We can do this by training a biLM on an unsupervised task and taking a linear combination of its internal state vectors as the word embedding.

4 Related Work

Word2Vec and GloVe are two popular word embedding techniques. People often use subword embeddings to provide extra information (e.g. if the word has not been seen often or at all). We use character-level convolution for this purpose.

Other work, like context2vec and paragraph vectors consider a word’s context.

Prior work shows (and our work confirms) that lower layers of a biLM learn syntax while higher layers learn context.

5 ELMo: Embeddings from Language Models

ELMo embeddings are a function of an input sentence. We solve a language modeling problem, where we aim to predict the next (or previous) token given the current token and the ones before (or after) it. That is, we aim to maximize:

$$\sum_{k=1}^{N} \log p(t_k | t_1, ..., t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log p(t_k | t_{k+1}, ..., t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \tag{1}$$

We use L2 regularization when optimizing this objective function.

Each layer has a forward and backward LSTM, whose hidden states we concatenate and pass to more forward/backward LSTM layers. This gives a set of vectors for token k:

$$\begin{aligned} R_k &= \{x_k^{LM}, \overrightarrow{h}_{k,j}^{LM}, \overleftarrow{h}_{k,j}^{LM} | j = 1, ..., L\} \qquad (2) \\ &= \{h_{k,j}^{LM} | j = 0, ..., L\} \qquad (3) \end{aligned}$$

where $h_{k,0}^{LM} = \mathbf{x}_k^{LM}$ and $\mathbf{h}_{k,j}^{LM} = [\overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM}]$ for $j = 1, ..., L$. To turn these into a single word vector, we use

$$ELMO_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM} \qquad (4)$$

where the entries of $\mathbf{s}^{task}$ sum to one and $\mathbf{s}^{task}$ and γ^{task} are learned.

To use this in another task, you can use the ELMo vectors as the word embedding, or concatenate them to your existing word embedding, or concatenate them to the vector that goes into your final few layers.

6 Evaluation

We evaluate on question answering (SQuAD), Textual entailment (SNLI), Semantic role labeling (OntoNotes), Coreference resolution (OntoNotes), Named entity extraction (CoNLL), Sentiment analysis (SST-5). We set state of the art on each one.

7 Analysis

Using the final ELMo layer gives most of the gains, but you can get a little extra if you use them all.

We experiment with different L2 regularization parameters λ.

Usually, we use ELMo at the start of the neural net for whatever task we solve. However, if you put it at both the beginning and the end, you may get a little extra gains.

Looking at nearest neighbors for some sentences, it seems that ELMo figures out polysemy (e.g. it can tell whether play refers to playing a game or to a theatrical performance).

We use ELMo for a word sense disambiguation task and find that the final layer vectors are competitive with state of the art, which shows that they are learning context.

We use ELMo for a part of speech tagging task and find that the lower layer vectors are better than the final layer, which shows that they are learning syntax.

We compare ELMo to CoVe and find that ELMo is better.

ELMo increases sample efficiency, so you need fewer iterations and less training data to get good performance on your task.

Looking at the learned task-specific weights for the biLM vectors, we find that placing ELMo at the beginning of the model prefers earlier biLM layers while placing it at the end has about equal preference over all biLM layers.

8 Conclusion

ELMo uses a biLM to learn word vectors given the sentence containing the word. These word embeddings are very good in other NLP tasks.