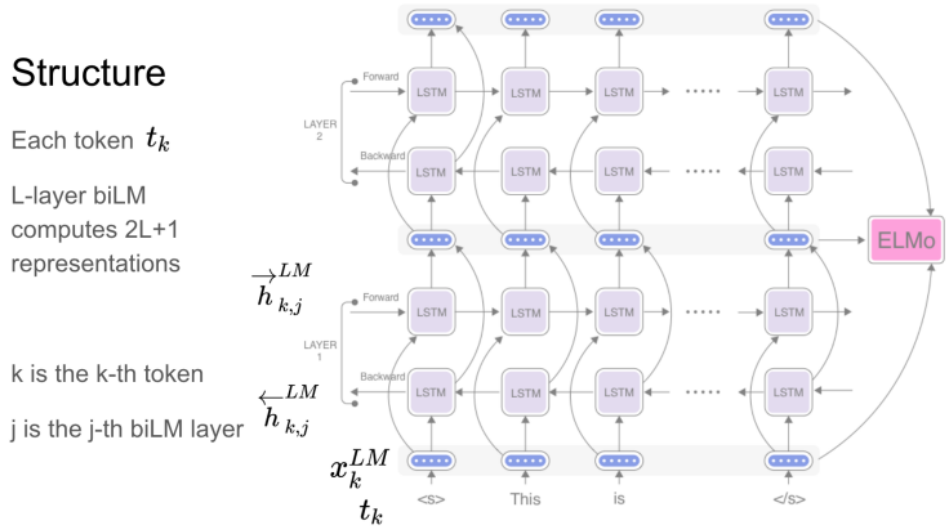
"Wir müssen wissen — wir werden wissen" — David Hilbert

ELMo in Practice

ELMo: **Deep contextualized word representations**
In this blog, I show a demo of how to use pre-trained ELMo embeddings, and how to train your own embeddings.

Features?

- Pre-trained Embeddings from Language Models. Structure: Char-based CNN and Bidirectional LSTM (any number, 2 is typical).
- Content-dependent word representations. Given the same word, the embeddings for it may be different! Depending on the sentences and contents.
- Improved on supervised NLP tasks including question answering, coreference, semantic role labeling, classification, and syntactic parsing. Before **BERT**. But BERT you may need TPUs… ELMo is somehow enough for your own corpora. lol



Structure?

A nice illustration from **this blog.** It has two Bi-LSTM layers and a lower level word embedding layer.

How to use?

There are many ways for you to use the pre-trained embeddings (from the previous figure).

- Collapse into R-dim. You can concatenate all the three layers and make it a very large/long vector.
- Learn task-specific weights. In the original paper, they learned a weight for each of the three layers. Then applied a weighted sum:

$$\text{ELMo}_k^{\text{task}} = E\left(R_k; \Theta^{\text{task}}\right) = \gamma^{\text{task}} \sum_{j=0}^{L} s_j^{\text{task}} \mathbf{h}_{k,j}^{LM}$$

- Simplest (top level).
- Train your own embedding and concatenate with ELMo embeddings. If you have your own corpus, you can train your own 'static' word embeddings like word2vec. Then concatenate your 'static' one with ELMo embedding.

Train your own ELMo?

There may be various methods and links. But the following is the route for myself. The **code** for training is in TensorFlow. It is a CUDA version. When you install TF with GPU, be careful to check if your CUDA version is compatible with the TF version. Once the installation is complete. You need to prepare the following files to train:
– A vocabulary file.
– Training file: each row contains a raw sentence.
– Validation file: same formate with training file.

When start training, it is essential to check the printed perplexity. Terminate at 40 or more is fine. Rember to validate the model.
I tried with ~2m sentences, <24h on 5 GPUs. My training perplexity was 40+, and in the validation file, it was about 30+.

If you want to continue importing the embedding using pytorch, you will need to convert into .hdf5 files.

Code?

Simple test **code** from Github.

Share this:

Like

Be the first to like this.

Related

TensorFlow 07: Word Embeddings (2) - Loading Pre-trained Vectors
January 17, 2017
In "Python"

NLP 05: From Word2vec to Doc2vec: a simple example with Gensim
July 27, 2016
In "Algorithm"

Deep Learning 19: Training MLM on any pre-trained BERT models
March 28, 2021
In "Theory"

Irene December 17, 2018 Deep Learning, Python, PyTorch

Published by Irene

Keep calm and update blog. **View more posts**

2 thoughts on “ELMo in Practice”

Maryam hammami
April 4, 2019 at 8:31 pm

hello,
i would like to ask you if the 30+ is the AVERAGE PERPLEXITY?.

★ Like

Reply

Irene
April 5, 2019 at 3:44 pm

Yes, I think so.

★ Like

Reply

Leave a Reply

Enter your comment here...