Unsupervised Language Learning - Practical 2

Nuno Mota - 11413344, Tom Pelsmaeker - 10177590

May 19, 2018

In this report and implementation[a] we focus on 3 different embedding models, namely: SkipGram [7](SG), Bayesian SkipGram [1](BSG) and EmbedAlign [8](EA). We evaluate their performance on a Lexical Substitution Task(LST) and Alignment Error Rate (AER), for EA.

Methods

Details specific to our implementation of the models will be discussed shortly below. We refer to the papers for a more thorough discussion.

SkipGram

The training objective of SG is predicting context given central words, yielding center word encodings as embeddings. SG was implemented with negative sampling (NEG) [2] as softmax approximation, for faster optimization.

Bayesian SkipGram

BSG predicts word embeddings as a posterior densities given context, allowing the model to encode multiple meanings of a single word. BSG was implemented with the approximated log-partition function (JI) ([1] section 2.5).

EmbedAlign

EA exploits equivalence through translation as a form of distributional context, embedding words as posterior densities encompassing both languages. EA was implemented with both complementary sum sampling (CSS) and softmax (SM). We used a BiRNN as encoder, with GRUs as recurrent units.

Experiments

All models were trained with the Hansards dataset. We did not subsample frequent words nor reduced the vocabulary size. However, all tokens were lowercased.

For SG and BSG we used a context window of 5, padding when necessary, and sampled, according to unigram statistics raised to $\frac{3}{4}$, as many negative context words as we had positive ones (k=1).

For EmbedAlign we only kept sentence pairs for which both sentences had 30 words or less, padding smaller sentences.

All models embed into 100 dimensions, with the GRU hidden states, of EA, having a dimensionality of 100 also, following [8].

We used the Adam Optimizer [4], minibatches of size 1024 and an adapted learning rate of 1e-2 according to the linear scaling rule described in [3], in all reported results.

We trained our models until convergence on the training set without early stopping, since validation loss is not a good predictor of task performance [5]. EA could have been stopped based on validation AER, but we chose not to do so for consistency with the other models. For EA we annealed the KL term, from 0 to 1, with 1e-3 increments every mini batch. The results were not significantly influenced by different hyperparameter configurations, like batch size, learning rate, and convergence threshold.

Results

All our models were evaluated on LST using the provided scripts. For all the models we estimate substitutions based on cosine similarity (computed with the posterior mean for BSG and EA) and, when possible, also based on KL divergence (Table 1).

For EA we also report the AER with alignments based on $\max\left(P\left(y_j, a_j|\mu_1^m\right)\right)$, $\mu_1^m = \mathbb{E}[Z_1^m]$ (Table 2).

Model	cos	KL
SG_{NEG}	0.330	-
BSG_{JI}	0.276	0.295
EA_{CSS}	0.293	0.261
EA_{SM}	0.282	0.283

Table 1: English GAP on LST test data.

Model	AER
EA_{CSS}	0.854
EA_{SM}	0.851

Table 2: EN-FR AER. These scores indicate an unsolved bug in either the AER script or embedalign.

Conclusions

It can be concluded that none of the models perform well on the LST, with only SG outperforming the random baseline of 0.3 [6]. This is most likely due to the small size of the Hansard corpus and the lack of rigorous preprocessing, if not due to mistakes in our implementation. Next to this, it could be that having more data would allow the more expressive models to learn richer representations (read posteriors), eventually surpassing SG, as SG embeddings would be hindered by being point estimates.

[a] `https://github.com/0Gemini0/ULL/tree/master/Practical2`

References

[1] Arthur Bražinskas, Serhii Havrylov, and Ivan Titov. Embedding words as distributions with a bayesian skip-gram model. *arXiv preprint arXiv:1711.11027*, 2017.

[2] Yoav Goldberg and Omer Levy. word2vec explained: Deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

[3] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[5] Siwei Lai, Kang Liu, Shizhu He, and Jun Zhao. How to generate a good word embedding. *IEEE Intelligent Systems*, 31(6):5–14, 2016.

[6] Oren Melamud, Omer Levy, and Ido Dagan. A simple word embedding model for lexical substitution. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 1–7, 2015.

[7] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[8] Miguel Rios, Wilker Aziz, and Khalil Sima'an. Deep generative model for joint alignment and word representation. *arXiv preprint arXiv:1802.05883*, 2018.

Note on AER

As can be seen in Table 2, EA is unable to produce alignments that are even remotely convincing. This is most likely to a bug somewhere in our implementation. We tried a bunch of different ways to train EA, varying KL annealing, batch size, learning rate and training/prediction with CSS or SM, but none of it influenced the AER significantly. Next to a bug, another reason for the bad AER could be the unavailability of a NULL word to align to. However, as the focus of this assignment was on word embeddings and not on alignments per se, we did not include such NULL words in our data preprocessing pipeline. Even though it is unclear whether our EA implementation is broken on LST also, it is unlikely that a much higher score would have been obtained with the small hansards dataset. Thus we view the conclusions section valid regardless of whether EA was implemented correctly or not.