From Word Embeddings to Natural Language Definitions

Ankita Pasad
Toyota Technological Institute at Chicago
ankitap@ttic.edu

Abstract

The effectiveness of distributed word embeddings in word similarity and analogical relation tasks shows their capability to capture the semantic information. This notion of meaningful information contained within a fixed length vector can be more directly evaluated by its performance on the task of generating definitions given the word embeddings. Inspired from a previous related work on definition modeling, this work tries to solve and analyze the same task with a less constraint model and lesser data. A gated recurrent neural network based architecture has been presented for this task with multiple evaluation metrics in place. As it is hard to evaluate the performance of generative tasks because of their subjectivity, the results show that although the model learns to generate meaningful sentences as the definition, the evaluation metrics used to quantify the success of the model do not show a very promising performance.

1 Introduction

In the past half a decade, several high-quality word embedding learning neural network techniques techniques have been introduced (Mikolov et al., 2013a) (Yogatama et al., 2015) (Pennington et al., 2014). Following that these embeddings have been used for several tasks and have been seen to capture syntax and semantics. For instance, the performance of embeddnigs on analogical relationship tasks gained a lot of popularity (Mikolov et al., 2013b). However these tasks do not evaluate the embedding's lexical information as directly as a definition modeling task, for instance, would. So we study whether these pre-trained word embeddings can generate natural language English definitions of their corresponding words. The models designed for this task can be used to evaluate the semantic quality of word embeddings and the short-comings.

Formally, **definition modeling** is defined as "the task of estimating the probability of a textual definition, given a word being defined and its embedding" by Noraset et al. (2017). The model is trained on a subset of a dictionary (word-definition pairs) and then is evaluated on the task of generating definitions of new words, could be also seen as a task of extending an existing dictionary. Given the nature of the definitions, more often than not only a few words are specifically related to the word being defined, the rest being common function words. In order to capture this the word being defined is provided as an input to every time step.

In the next section we discuss the related work (Noraset et al., 2017) in greater detail. Section 3 has details on the model architecture used. Dataset specification, training specific details and evaluation metrics have been discussed in 4. Section 5 shows the observed results and the analysis. Next we conclude with the future work in section 6.

2 Previous Work

This work has been largely inspired by the work on Definition Modeling by Noraset et. al. (2017). Their proposed model works on 300 dimension Word2Vec pre-trained embeddings as an input to a two layer Long Short Term Memory Networks. This work also experiments with other word-level features - affixes and hypernym embeddings, learned using convolutional neural network architecture and concatenated with the input word-embeddings. Multiple metrics are used for evaluation - perplexity, BLEU, and also one based on the work on reverse dictionary by Hill et al.

(2015), i.e how well are the words ranked for given definitions. It checks whether the reverse dictionary model can output the word embedding sufficiently close to the original word embedding given the definitions generated by their proposed model.

Our model differs in 1) there is no heavy feature engineering part involved at the input, i.e., we do not use any other word-level features, 2) the embedding vectors for the words in the definition are learned from scratch as opposed to using a fixed set of pre-trained vectors, 3) apart from BLEU and perplexity we use a classification-based evaluation metric, discussed in detail in section 4.3. Since we are working with a smaller dataset, there is no direct quantitative comparison possible.

3 Model Architecture

This is a sequence generation task where the input is a fixed dimensional feature vector, v^*, for the word being defined, w^*. The length of output sequence consisting of the words in the definition, $w_1, ..., w_k$ is not fixed. So the modeling problem we are dealing with here is $p(w_1, ..., w_k|w^*) = \prod_{i=1}^{k} p(w_i|w_1, ..., w_{i-1}, w^*)$. These conditional probabilities are modeled using a gated recurrent unit (GRU) (Cho et al., 2014) whose hidden representation h_t at time-step t can be written as an output of the recurrent non-linear function, $h_t = g(v_{t-1}, h_{t-1}, v^*)$, where v_{t-1} is a learned embedding representation of the word w_{t-1}. Note that according to this model, the seed v^* is given as an input at every time step. Formally, ($\hat{v}$ is the learned embedding vector transformed through a non-linear layer)

$$\begin{aligned}
h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot q_t \\
q_t &= \tanh(W[v^*, \hat{v}_t] + U(r_t \odot h_{t-1})) \\
z_t &= \sigma(W_z[v^*, \hat{v}_t] + U_z h_{t-1}) \quad \text{(update gate)} \\
r_t &= \sigma(W_r[v^*, \hat{v}_t] + U_r h_{t-1}) \quad \text{(reset gate)}
\end{aligned}$$

At the output of every GRU unit, a softmax layer is used to convert the output scores to a probability distribution. The model is trained on the cross-entropy loss function.

4 Experimentation

4.1 Dataset specifications

GloVe (Pennington et al., 2014) pre-trained word embeddings have been used to represent the words being defined. These 300 dimensional word vectors were trained on the common-crawl web database. Of the 2.2M words, only the 50,000 most frequent words observed while training these embeddings were retained with their corresponding vectors. This helped in limiting the size of the dataset for faster computation and also ensured that the word embeddings we are using have been trained on a significant amount of dataset so as to have a semantically meaningful representation. The open-source tool, Chakin[1], was used for downloading these pre-trained word embeddings.

For the word-definition pairs, the dataset made publicly available[2] by Noraset et. al. (2017) is used for this work. It is a compilation of parsed dictionary entries from GCIDE[3] and WordNet's glosses. The distribution of the effectively available dataset size, after limiting the word embedding list to 50,000, is provided in table 4.1. Each of train, development and test set can possibly have multiple entries for same words with different definitions, but there is no overlap of words between these sets. The dataset also has information about the part of speech (PoS) tag of the defined word and the source dictionary of the word (GCIDE/WordNet). The average number of definitions per word is ≈ 6.6.

For the second set of experiments we use the PoS tags as well. Every input now is the word to be defined along with its part of speech tag. The embeddings for PoS tags are learned through the model. There are about 40 PoS tags in the dataset, but we club them together into 5 tags (which are then learned into a 5-dimensional embedding vector) which roughly correspond to - noun, verb, adjective, adverb, and preposition, conjunctions and others are clubbed into 1 tag.

Number of	train	dev	test
word-definition pairs	89,449	5,082	5,186
distinct words	9608	547	548

Table 1: Dataset statistics

4.2 Training

Traditionally, while training a recurrent neural network (RNN) for decoding previous ground truth output is passed as an input to the next time step, this is called teacher forcing. However, during inference the model uses previous unit's output as an

[1]https://github.com/chakki-works/chakin
[2]https://github.com/Websail-NU/torch-defseq/tree/master/data
[3]http://gcide.gnu.org.ua/

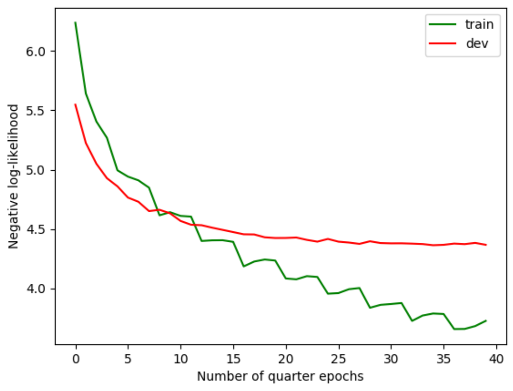

Figure 1: Loss convergence during training for model without PoS tags

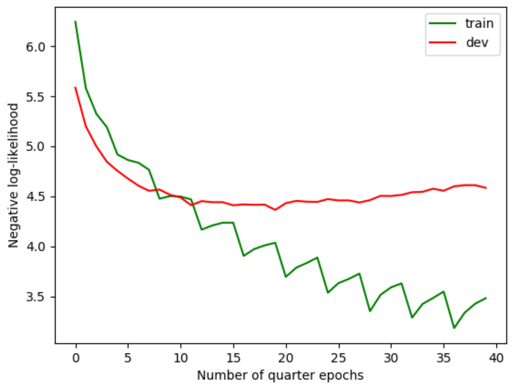

Figure 2: Loss convergence during training for model with PoS tags

input to the current unit. This discrepancy between train and test time could lead to cascading errors during inference since the model has never been trained on the model output as an input scenario. In order to deal with this scheduled sampling was proposed by Benjio et al. (2015). During training, at each time step, the model uses the groundtruth as an input with the probability set by the teacher forcing ratio (tf) and selects the previous decoded token with probability (1 - tf).

For both set of experiments we use a 2 layer GRU network with 300 dimensional word embeddings and hidden layers. Mini-batch size of 50 is used with the teacher forcing ratio of 0.8. Adam optimizer with a learning rate of 0.0015 is observed to perform the best. Early stopping criteria is used to stop training and it is observed to stop at around 4-6 epochs. Greedy decoding based approach is used during inference. The training and validation loss has been shown in figure 1 and 2. We can see that the inclusion of PoS tags also helps increase the rate of convergence.

4.3 Evaluation Metrics

Since the dataset used is different from that in the related work (Noraset et al., 2017) because of limited number of word embeddings being used, it is difficult to do a fair comparison with the baseline. Along with the qualitative analysis of the generated definitions, three different evaluation metrics, have been used to quantify the performance of this task.

4.3.1 Modeling task

Perplexity on the corpus-level has been reported by finding how well the proposed architecture models the definitions from the held-out dataset (teacher-forcing ratio of one). The total loss for all the words in the test corpus is averaged to get an average modeling loss L and 2^{-L} is reported as perplexity. For the sake of comparison, the perplexity on the train set and validation set has been reported too.

Classification-based metric has been defined to have a more intuitive interpretation of how discriminative the model is when it comes to ruling out an incorrect definition. For every word-definition pair in the help-out set, 10 “incorrect” definitions are sampled and the probability that the model gives to each of those definitions with the same seed (word embedding) is evaluated. The accuracy of how often does the model give highest probability to the correct definition is reported. For a baseline, all the 11 sentences are converted into their respective sentence embedding vectors by averaging the individual word embeddings from the same GloVe corpus. Accuracy is calculated as how often does the closest sentence embedding to the embedding of the word under consideration correspond to the “correct” sentence.

4.3.2 Generation task

For the definitions generated by the model (teacher-forcing ratio of zero) on the held-out dataset, **BLEU** (Papineni et al., 2002) scores on the corpus-level have been reported using the natural language toolkit in Python (Bird and Loper, 2004). As a baseline, for all the words in the train corpus having multiple definitions one definition is used as hypothesis and the rest are used as reference, and a corpus-level BLEU score is reported. The baseline can be considered as human-level performance on the task since we are comparing one gold-standard against other gold-standards.

5 Results and Discussion

The results have been tabulated in tables 1 and 2. From the perplexity and BLEU values we can see that adding PoS not only helps better modeling but also leads to better quality of generated definitions.

The results on the classification-based metric are disappointing since the baseline is always better than the performance on both train and test. On closer look at the loss values of the actual and negative examples it seemed like even when a negative example had best loss, the loss for actual definition was not too high. So I used 2 metrics - top 1 and top 3, where the latter checks whether the actual definition lies amongst the least 3 loss values. Although the top 3 performance is better than the top 1, the results are still poorer than the simple baseline model. This implies that - 1) sentence embedding obtained by averaging the word embeddings actually retains the meaning of the word very well, and 2) even though the model gives a high likelihood to the actual definition (as can be seen from the perplexity values), with a good chance there is another definition in the negative example set which gets a higher probability. Looking at the actual examples more closely does not reveal anything insightful, for instance, for the word 'jointly' (actual definition: 'not separately'), the definition 'pretended' gets a lower loss. It does not even have to do anything with the length of the sentence, for instance - the word 'labor' (actual definition: 'servile toil') picks a definition which has 30 words instead, none related to the word 'labor'.

The generated definitions with PoS tag inputs are interesting. For example for the word 'light' which appears with 8 different tags in the test set, a verb tag generates the definition 'to make feverish', adverb tag generates the definition 'slightly' and noun tag generates the definition 'capable of being seen'. Thus showing that even though the PoS embedding vector is as small as 5-dimensional, it is capable of learning syntax specific information. The model A generates just the word 'to' as a definition for 'light'. It does generate interesting definitions too, for instance, 'creek' is defined as 'a large mass of land or a rivers' which is interesting because parts of these definition appear in very different context in the train set, but the model learns to join them in a grammatically and semantically correct way. Another such example is the word 'mathematical' which has the definition 'involving algebra'.

Metric \ Dataset	Train A	Train B	Test A	Test B
Perplexity	4.48	4.37	7.23	6.9
BLEU (%)	9.8		5.2	6.8

Table 2: Quantitative comparison of results using different evaluation metrics; A: without PoS tags, B: with PoS tags

Evaluation on		Classification-based (%) top 1	top 3
Train	A	42	64
	B	26	64
Baseline	A	54	72
	B	52	70
Test	A	24	48
	B	21	48

Table 3: Quantitative comparison of results using 11-way classification; A: without PoS tags, B: with PoS tags

6 Future Work

Even though the model generate some interesting definitions, it still fails to generate even meaningful sentences as an output for some words. In place of greedy decoding, beam search or sampling based decoding could help solve this issue since it gives more flexibility.

One drawback of this method is that we need to have a word-embedding of the word being defined. In order to circumvent this, we could have a character-level encoder-decoder model with attention in place of using a fixed pre-trained word embedding. The attention will also help us visualize the whether different parts of the generated definitions are giving particular weightage to different morphemes.

Acknowledgements

I would like to thank Prof. Karen Livescu and Prof. Kevin Gimpel for their suggestions and involvement throughout the course of the project.

References

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179.

Steven Bird and Edward Loper. 2004. Nltk: the natural language toolkit. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 31. Association for Computational Linguistics.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. 2015. Learning to understand phrases by embedding the dictionary. *arXiv preprint arXiv:1504.00548*.

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

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751.

Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, pages 3259–3266. AAAI press.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Dani Yogatama, Manaal Faruqui, Chris Dyer, and Noah Smith. 2015. Learning word representations with hierarchical sparse coding. In *International Conference on Machine Learning*, pages 87–96.