A Report: Paraphrase Embeddings

Student: Yiyi Chen, Dozent: Dr. Benjamin Roth

LMU Muenchen, CIS

Abstract. This is a report of a natural language processing project of learning word embeddings for syntactic path dependency between entities. First, we will give an introduction to the project; then, the dataset and the methodologies are illustrated; finally, the experiments, results and observations are presented.

Keywords: word embeddings · syntactic dependency paths · entities.

1 Introduction

The task is to learn embedding vectors for syntactic dependency paths between entities. In this project, we use three main methods: word2vec, universal schema and sequential model.

2 Dataset

The given data "word2vec_train_untyped.txt" has instances as following :

```
arg0:m.0_00  dobj^-___accuse___nsubj  arg1:m.07q13
```

in which, `arg0:m.0_00` is one entity and `arg1:m.07q13` is another, and `dobj^-___accuse___nsubj` is the dependency path between the two entities.

In order to train the word embeddings of the dependency path between the two entities, we take the entity pair as one token (using "#" to concatenate the entities), and the dependency path as another as following:

```
m.0_00#m.07q13  dobj^-___accuse___nsubj
```

Eventually, "test_simplified.csv" is used to evaluate the trained word embeddings of the dependency paths, for example:

```
~0~-dobj^-___assure___nsubj-~0~,~0~-dobj^-___accuse___nsubj-~0~,no
~0~-dobj^-___assure___nsubj-~0~,~0~-dobj^-___convince___nsubj-~0~,yes
```

The first line states that the dependency path `dobj^-___assure___nsubj` does not imply the dependency path `dobj^-___accuse___nsubj` (i.e., a negative sample) , while the second line states that `dobj^-___assure___nsubj` implies `dobj^-___convince___nsubj` (i.e., a positive sample). There are in total 1717 unique dependency paths, 2661 negative samples and 1322 positive samples.

The data "word2vec_train_untyped.txt" is filtered so that the data we used for experiment only contains the paths that occurs at least once in "test_simplified.csv". Then, in order to train the data using enough negative samples (we experimented on negative factors 2,3 and 5, then we decided on negative factor 5 for the best result), we further filter the data so that it contains entity pairs which occur at least three times.

For each dependency path and entity pair, we sample 5 negative entity pairs. In order to save RAM, the data is structured as following (aligning the dependency path, positive entity pair and negative entity pair):

```
dobj^-___join___nsubj m.0_00#m.01gf5z m.05cgv#m.0d0516
dobj^-___join___nsubj m.0_00#m.01gf5z m.0dtj5#m.045c7b
dobj^-___join___nsubj m.0_00#m.01gf5z m.01mrvg#m.044qx
dobj^-___join___nsubj m.0_00#m.01gf5z m.0g0c_#m.0gwlg
dobj^-___join___nsubj m.0_00#m.01gf5z m.03qpbs#m.01xyt7
```

The processed dataset has following statistics:

Table 1. The statistics for dataset

Entities	Entity Pairs	Paths	Samples	With negative samples
19396	66835	1714	410370	2051850

Because the data is very large, we use 99% (2031332 samples) of data for training and 1% (20518 samples) of data for validation.

3 Methodology

Word2Vec Model In this model, the objectives $sigmoid\{pos\_ents^\intercal path\} \sim 1$ and $sigmoid\{neg\_ents^\intercal path\} \sim 0$ are used [1].

Universal Schema Model In this model, the objective $sigmoid\{pos\_ents^\intercal path - neg\_ents^\intercal path\} \sim 1$ is used [2].

Sequential Model In this model, the objective $sigmoid\{pos\_ents^\intercal path - neg\_ents^\intercal path\} \sim 1$ is used, since the result from Universal Schema Model is better than Word2Vec Model.

4 Experiments & Result

Hyperparameters For all the experiments, the batch size is 4096. We experiment with different numbers of epochs (20, 30, 40 and 80), optimizers SGD, Adadelta and Adam (Amsgrad) with learning rate 0.001, 0.01 and 0.1. For the Sequential Model, we experiment with LSTM, GRU and CNN1D for hidden layers. For all

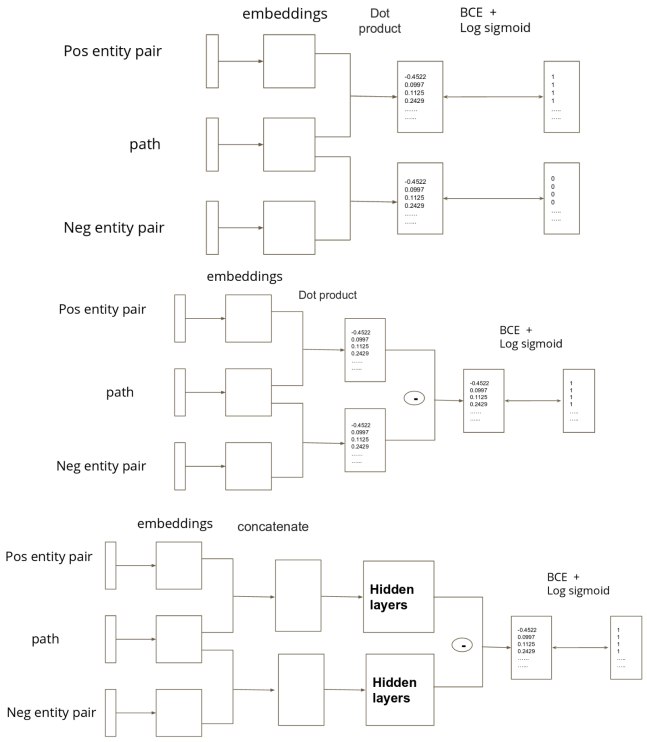

Fig. 1. Word2Vec Model, Universal Schema Model and Sequential Model in order

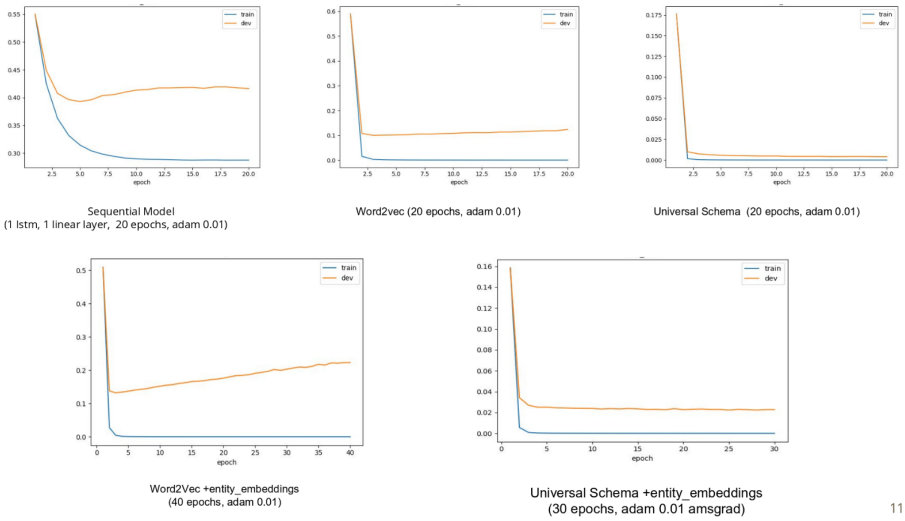

Fig. 2. Loss graphs for model training

the models, we also experiment with activation functions ReLU and Tahn. For Universal Schema and Word2Vec Models, we also initialize the embedding vector separately for entities and add them to the embedding vector for the entity pair.

Result The Fig. 1 shows the basic structures of the models used in this project, and the Fig. 2 shows the loss graphs for training the models. See Table. 2 for the best result from each model with its hyperparameter settings.

	Mean Precision	Mean Recall	Mean F1 Score
Baseline	0.471	0.712	0.565
Word2Vec (20 epochs, Adam 0.01)	**0.566**	0.720	0.633
Word2Vec + entity_embeddings (40epochs, Adam0.01)	0.518	0.777	0.621
Universal Schema (20 epochs, Adam 0.01)	0.549	0.787	**0.646**
Universal Schema + entity _ embeddings (30 epochs, Adam 0.01 Amsgrad)	0.507	0.796	0.618
Sequential Model (30 epochs, Adam 0.01 Amsgrad)	0.418	**0.809**	0.551

Table 2. Result of testing embedding vectors of dependency paths on "test_simplified.csv"

5 Observations

Data Structure Data samples structured as `path pos_entity_pair neg_entity_pair` is much less memory consuming than `path pos_entity_pair True` and `path neg_entity_pair False`.

RAM To report the `dev_loss`, detach the tensor and only record the number `loss_dict["dev"].append(dev_loss.detach().item())` to avoid `out of memory` error.

Embeddings Concatenating the embeddings of `entity1` and `entity2` in order to get the embeddings of `entity1#entity2` does not work well for training the models, it leads to better results to use embeddings of `entity1#entity2` directly or embeddings of `entity1#entity2` plus embeddings of `entity1` and embeddings of `entity2`.

Unexpected Phenomenon The model with hidden layers does not work better than word2vec and universal schema, maybe due to the very short sequence length 2 (`path pos_entity_pair` or `path neg_entity_pair` as one sequence).

Optimizer The Universal Schema Model with SGD optimizer does not work very well as in [2], rather, all the models have better results with optimizer Adam.

Activation Function Although the models with ReLU function leads to lower loss values, they do not render better results for dependency path embedding vectors.

Conclusion For this project, the simpler models without activation function have better results (Universal Schema and Word2Vec models compared to Sequential Model with hidden layers). And, Universal Schema with its ranked pair objective has better performance than Word2Vec on F1 score.

Future Work It might be worth of investigating more in sequential model with ranked pairs objective on data having richer latent features. One reason for the better performance of simpler models in this project might be that the data does not have very rich features to exploit.

References

1. Tomas Mikolov, Kai Chen, Greg Corrado & Jeffrey Dean: Efficient Estimation of Word Representations in Vector Space (2013).
2. Sebastian Riedel, Limin Yao, Andrew McCallum & Benjamin M. Marlin : Relation Extraction with Matrix Factorization and Universal Schemas. In: Proceedings of NAACL-HLT 2013, pages 74-84.