Name: Kenny Patel

Roll No. : 19074010

Lab Task 7: Bilingual Dictionary Induction from Comparable Corpora

Introduction

The topic of the task is bilingual lexicon induction from comparable corpora. Its aim is to extract for each given source word, its target translations. To extract bilingual lexicons from comparable corpora, a well-known word embedding approach that maps source words in a target space has been used.

Task

Work on one language pair in both directions (English-German and German-English):

1. Using Word2Vec(Skip-gram) embedding and the VecMap library, generate a bilingual dictionary.
2. Using fastText embedding and the VecMap library, again do the same and compare the results with the above implementation.

Directory Structure

The main folder of this repo contains two Jupiter Notebooks, which hold the entire implementation of the task.

The Corpora, Embeddings, and Models are quite huge so attaching the drive link where I have uploaded them all.

Datasets: (Present in the main folder) Drive Link for Data and Models

- fastText embeddings of English and German
- word2vec (SkipGram) embeddings of English and German in English_W2V and German_W2V folders respectively
- Training data for English-German and German-English
- Testing data for English-German and German-English

Models: (generated Inside the vecmap folder)

Procedure

- Downloaded the WaCKy training and testing corpora for English and German from here
- Got the fastText and Word2Vec Skipgram embeddings from their respective links
- Cloned the Vecmap library to generate bilingual dictionary from here
- In order to build cross-lingual word embeddings, we will first train monolingual word embeddings for each language using fasttext and then map them to a common space with VecMap software. Having done that, we can evaluate the resulting cross-lingual embeddings using repo's included tools. Same goes for Word2Vec.
- The two main commands used for this are (these two are for german-to-english using fastText embeddings):

```
→ python3 map_embeddings.py --supervised ../Train_ger_eng.txt ../german.vec
../eng.vec SRC_MAPPED.EMB TRG_MAPPED.EMB
```

```
→ python3 eval_translation.py SRC_MAPPED.EMB TRG_MAPPED.EMB -d
../german_to_eng_dict_multiple.txt --retrieval csls --cuda
```

- Similarly, we will do for word2vec and then the same process in the other direction, i.e. English-to-German

Result

1. Using SkipGram

Language Pair	Accuracy (%)	Coverage (%)
German-English	63.90	81.30
English-German	54.9	90.30

2. Using fastText

Language Pair	Accuracy (%)	Coverage (%)
German-English	69.94	100.00
English-German	75.56	94.10

Conclusion

- The main difference between FastText and Word2Vec is the use of n-grams.

- Word2Vec learns vectors only for complete words found in the training corpus. FastText, on the other hand, learns vectors for the n-grams that are found within each word, as well as each complete word.
- This n-gram feature of FastText library solves the Out Of Vocabulary issue.
- Hence, we can observe higher accuracy when FastText embeddings were used as compared to Word2Vec embeddings.
- But due to the same feature of FastText library, the time taken for training and testing was more than Word2Vec(SkipGram).