COMP6714 Project 2 - Adjective Centric Embedding
Project Report

Boshen Hu
z5034054

1 Abstract/Summary

This report investigates an approach to improving upon the word2vec skip-gram model used for synonyms of adjectives. The result has also been demonstrated and discussed in this report.

2 Introduction

The word2vec model has been an important breakthrough for natural language processing (NLP). However, the vanilla word2vec model may not performed well enough on a specific problem, such as finding synonyms of adjectives. This could be because of the agnostic property of the model, i.e. the model does not care whether two words is in the same part of speech when it estimates the similarity of the words. As a consequence, the word2vec model can usually draw a discouraging conclusion that two words in different part of speeches are synonyms. To improve this, to find synonyms of adjectives as a target, the training method of the word2vec model has been modified slightly. The modification is based on the skip-gram model, and mainly focus on how to generate the batches for the training process. This could contribute to a “taste” of the model: training process will concentrate on adjectives. In addition, the data processing work also takes effect, as well as parameter tuning in model training process.

3 Methodology

This approach is built on the word2vec skip-gram model, and the training corpus is in "BBC_Data.zip". For reading convenience, I will only illustrate the specialty of my work, as well as the difference against the original vanilla word2vec model.

i) **Data Processing**

The python SpaCy library is used to parse the corpus. It helps to figure out the part of speeches of the words in the corpus. There is a simple idea that a word and its synonyms are in the same part of speech. In order to improve the performance of finding the synonyms, in the data processing part, a measure can be considered is to reduce the word amount in other parts of speech (i.e. not adjectives). This is based on a naïve assumption: If the vocabulary is fixed, the more the adjectives are in the vocabulary dictionary, the greater the probability that our model chooses an adjective as a synonym of the selected adjective is. The extreme case would be all of the words in the vocabulary dictionary is adjective, which forces the model always chooses an adjective (because no other part of speech words in the vocabulary dictionary). However, this may not be suggested, since if only adjectives are left, the relationship of context could be drastically destroyed. It could be hard to find a "reasonable" synonym for an adjective, even all of our choices are in the right part of speech. It is truely a tradeoff in terms of word reserving. Therefore, my approach is to preserve almost all the adjectives and some other "meaningful" word type, such as nouns, verbs and adverbs. The SpaCy is used to filter out the other part of speech words, such as conjunctions, prepositions, punctuation and numbers. Additionally, some of verbs (like forms of have, forms do and forms of be), some of nouns (like possessive nouns), some of adverbs (wh-adverbs) and even some of adjectives (wh-determiner) are removed as well. Finally, 13686 unique words within 3723 adjectives are left. Therefore, my training and evaluation work are based on these words.

ii) Training Data Generation

The most significant difference between the vanilla skip-gram model and mine could be from this part of the work. The vanilla skip-gram model does not care about whether a token is adjective or not, which deviates from our goals. Since the other part of speeches words (i.e. not adjectives) are just assistance for maintaining the context relationship, the training process could be better to focus on only adjectives, and this can be achieved by always giving the model inputs which has an **adjective** center word. By doing this, the model could be more sensible to get a relationship between adjectives. Hence, training data always have adjectives as center words(inputs), and the surrounding words to be the target words(outputs).

Because any other part of speech word as a center word input is skipped, a single adjective could have more chance to fetch its environment information (i.e. the context relationship), skip window can be wider than usual (a standard vanilla skip-gram model). In fact, the skip mode has been changed totally, all index of occurred adjectives are recorded on a list, and when generating a batch, sample indices are randomly chosen from the list. I use the full surrounding of an occurred adjective, since our refined corpus may not be very sufficient. To prove the insufficiency of the material, I count the whole refined document. There are 62443 times that adjectives have occurred altogether. Let us make an assumption: if we set a batch size to 64, by 100,000 steps, we need exactly 6,400,000 input-output pairs. In average, an occurred adjective needs to generate about 103 (6400000 / 62443) samples. This indicates that, the corpus may not sufficient in this case, as well as the batch size could be set to a smaller number. Since I use full surrounding, the the skip_window is set to num_sample // 2. For script simplicity and validity of generating process, I always make sure batch_size is exactly divisible by num_samples so that when we calling generate_batch function, we choose (batch_size //

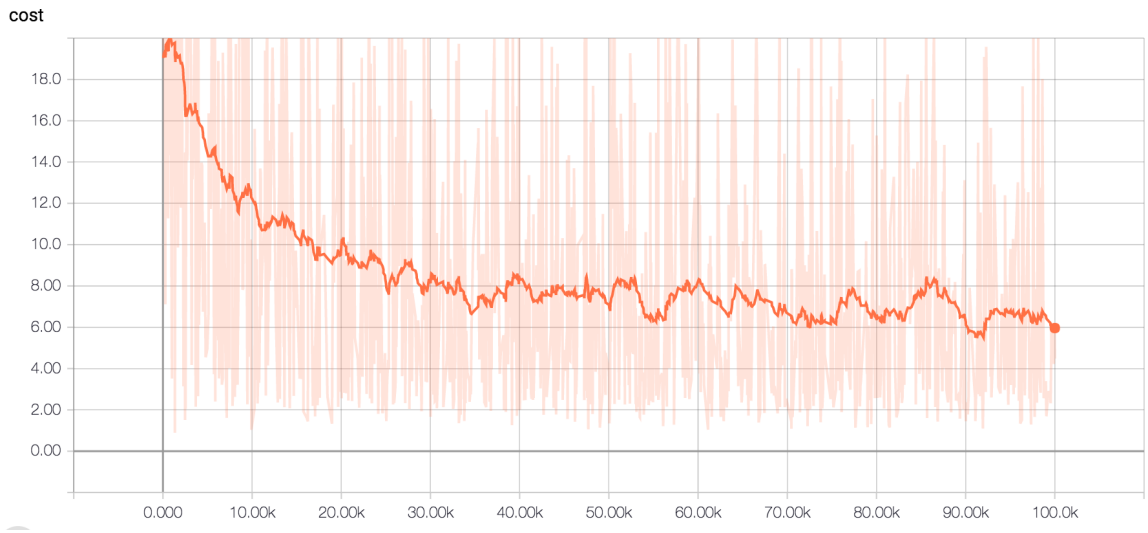
num_samples) adjectives to generate samples. Each sample can provide num_samples samples.

iii) Model Training

The model training work is almost the same as the vanilla skip-gram model. Tensorflow is used for training process. With the AdamOptimizer, learning rate has been tuned to 0.004, leading to a decent loss reduction.

4 Results and Discussion

With regards to different hyper parameter combination attempts, every single combination has been optimized by minimizing the loss. A representative loss figure has been demonstrated.



As the figure shows, the model has fully been optimized by the gradient descent method. The loss is eventually stable at 6.0 after 100000 steps. The next picture demonstrates an example of the model results of finding synonyms of the example adjectives(with hit numbers).

```
significant
['first', 'main', 'strong', 'high', 'major', 'good', 'grand', 'big', 'serious', 'basic', 'prime', 'great', 'chief', 'important']
14

single
['personal', 'several', 'different', 'available', 'same', 'clear', 'whole', 'only', 'original', 'simple']
10

special
['first', 'best', 'chief', 'major', 'own', 'private', 'different', 'particular', 'personal', 'several', 'extraordinary', 'main', 'vital']
13

specific
['sure', 'different', 'personal', 'only', 'simple', 'clear', 'certain', 'full', 'particular', 'own', 'right', 'local', 'final']
13

successful
['top', 'first', 'strong', 'good', 'full', 'better', 'big', 'major', 'high', 'important', 'popular']
11

top
['first', 'biggest', 'high', 'popular', 'chief', 'major', 'most', 'great', 'important', 'largest', 'better', 'main']
12
```

By using the provided evaluation file "*dev_set*", the average hits are computed as:

$$\frac{Total\ hits\ of\ all\ target\ words}{Total\ word\ amount}$$

The detailed results are demonstrated in the table.

Attempt Order	Average Hits	Batch Size	Skip Window	Num_samples	Vocabulary_size	Learning Rate	Number of Negative Samples
1	8.325	64	4	8	7000	0.004	10
2	8.275	64	4	8	13686(max)	0.004	10
3	8.625	32	4	8	7000	0.004	10
4	8.4	32	4	8	13686	0.004	10
5	8.6	8	4	8	7000	0.004	10
6	7.3	4	2	4	4000	0.004	10
7	7.875	32	4	8	10000	0.004	10
8	8.5	32	4	8	13686	0.0025	10

Table1

For comparison, a model is also trained by using the original vanilla skip-gram method.

Attempt Order	Average Hits	Batch Size	Skip Window	Num_samples	Vocabulary_size	Learning Rate	Number of Negative Samples
1	1.225	32	1	2	13686	0.004	20

Table2

With regards to performance,
it seems that no prominent differences in Table1 (even a lower vocabulary size may cause a slight higher average hit.), my solution has the average hit around 8.45.
However, when it comes to speed, given a smaller batch size as well as vocabulary size can lead a faster training steps.

Finally, although Attempt Order 3 in Table 1 has the best performance, I use the parameters of Attempt Order 8 in Table 1 in order to make sure that all the adjectives (existing in the raw corpus) are included in the embedding file.

Generally, my solution:

i) Pros
   Generally, it is easy to find that my model has hits on ground truth (common sense) of adjective synonym more times. Even not a reasonable result word is very likely to be an adjective one.

ii) Cons
   Although has improved, my model does not get great hit numbers as well. In addition, antonyms are still a significant problem in this model.

5 Conclusion

This report has elaborated an approach to improve the general skip-gram model performance in terms of searching for adjective synonyms. Although a concrete improvement can be seen in this way, some difficulties such as recognizing synonyms against antonyms could be not improved even slightly. There are still many problems in this subject waiting for a development. These issues could be enhanced by a model more than a skip-gram based model.