Task 1: Distributional Semantics -Minjun

For Task1, the main goal of it is to execute experiments of given steps and evaluate the performance of the distributional semantic approach. In this report I will explain the choices that have been made to execute the experiments with the conclusion of my optimal solution for the given task.

1. Choosing Skip-Gram model

The Skip-grams technique is an unsupervised learning method for searching word embeddings. This method detects word embeddings by training a classifier to assess the probability of a specific context word to appear given a target. Although, there is a similar method called CBOW(Continous Bag of Words). I have chosen Skip-Gram model instead of CBOW for the following reasons:[1]

Skip-Gram model is better than CBOW model in with small datasets which I think it is the case for the given dataset.Skip-Gram has the advantage over CBOW when the data has less frequent words. This is not a significant advancement, but product reviews use a variety of vocabularies, which, in my opinion, are crucial in determining how the product is presented to consumers.

The Skip-gram model architecture is made up of three layers: an embedding layer that matches each word to a vector of values, a linear layer, and a softmax function that converts the linear layer output to a probability distribution. The model's final probability distribution indicates the likelihood of each context word occurring given the input keyword. The weights of this classifier's embedding layer may be efficiently applied for word embeddings. Due to the similarity of the model to a linear regression with a softmax function applied to the output of the linear layer, it is a relatively flat mode, allowing for backpropagation to effectively be applied back to all layers, and has a comparatively modest number of weights. This decreased the time needed for the training.

2. Iterations and experiments

All of the experiments are taken with 5 iterations. The great number of iterations would give a better results, but the time taken to run the experiments were too long to run the code. The results for the experemients report the standard deviation and average accuracy. For every iterations, the reported results can have a little difference than before, because the reverse word corpus is randomly generated for every iteration.

3. Experiment on window-size

From the experiments, it is clear to say that the lower the window size, the higher accuracy of the results. The reason for this is smaller windows lead to a more syntactic representation, shorter window sizes capture syntax and longer ones capture semantics. In addition, I have run the experiments with stemming and removing stop word. Applying either technique decreases accuracy with smaller contexts. The experiments confirms that smaller contexts reflect syntax in representation.

4. Size of Embedding Dimensions

The results of the experiments show that the size of word embeddings: 200, 250, 300 provide the best accuracy for a context window size 1. Shorter word embeddings are less specific regarding keyword features. Longer embeddings over 300 may need more extensive training data to attain peak performance.

5. Complete-linkage Agglomerative Clustering

In this case, agglomerative clustering with complete linkage is used. From the comparison of the K-Mean and addlomerative clustering[2], it is significantly more accurate to use this method of clustering.

6. Stemming and stopword removal and conculsion

From the experiments, the accuracy falls when stopwords are removed. In my opinion, stopwords contains important information about the keyword. The highest accuracy of 91% has been achieved with a window size 1, embedding length of 300, no stemming, no stopword removal and agglomerative clustering with complete linkage. From this result, there is a benefit from syntactic representation being more suitable for this task and from the explicitness of longer length of embeddings deminsion.

---

[1] "Word Embeddings: CBOW vs Skip-Gram - Baeldung." 11 Nov. 2022, https://www.baeldung.com/cs/word-embeddings-cbow-vs-skip-gram. Accessed 19 Dec. 2022.

[2] "Comparing different clustering algorithms on toy datasets - Scikit-learn." http://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html. Accessed 19 Dec. 2022.

Task 2: Neural Network for Classifying Product Reviews

1. Model discription CNN vc LSTM

A wide range of neural network model designs have been demonstrated to be useful for sentiment analysis tasks. I decided to build a Convolutional Neural Network (CNN). There is a reason that I have chosen CNN rather than LSTM. An advantage of this in comparison to LSTM is that CNNs do not suffer from the limited attention information bottleneck problems that occur with LSTMs[3]. Due to this reason, for most sentiment analysis tasks, CNN models tend to outperform LSTM models.

In addition, 'Convolutional Neural Network: Text Classification Model for Open Domain Question Answering System'[4], gave me a clearer plan for the experiments.

In many cases, CNNs are designed to handle computer vision tasks. Because words are represented as embeddings in deep learning tasks, text can be represented as a two-dimensional matrix with concatenations of word embeddings similar form to an image. On this form, convolutional layers of size can be used to target n-grams with our convolutional layer. In my implementation, the embedding layer is applied to represent the sentiment text as a list of dense vectors. This list is inserted in to three CNN layers which have the filter sizes of (3 x embedding size), (4 x embedding size) and (5 x embedding size). This allows the neural network to infer on all tri-gram, four-grams and five-grams in each sentence. After that, each convolution is given its own pooling layer. The sentiment length influences the output dimensions of the convolutional layers.The pooling layers guarantee that the output of all convolutional layers is uniform in length regardless of text size. A fully connected layer and a sigmoid layer process this information to output a scalar in the range [0,1] to describe whether the model determines a text is negative or positive has a negative or positive sentiment. Each value less than 0.5 is considered negative and each inference greater than or equal to 0.5 is considered positive.

2. Experiments

Stopword removal had a negative impact on accuracy. The reason for that is that the nltk stopwords list includes tokens indicating negation such as "not". Because of this stemming also negatively affected the accuracy part of words such as "n't".

Dropout Handling overfitting is a difficult task with this dataset. On training data, accuracy of greater than 95% is frequently achieved, whereas evaluation accuracy is close to 70%. The model is struggling to generalise its inferences. The model was having difficulty generalising its inferences. To decrease the level of overfitting, I added dropout to the fully connected layer's input. Dropout is not applied to the testing data. From this, there was 2% accuracy improvement.

There are some texts that have both negative and positive tags. There are close to 50 that had such sentiments in the corpus. From my experiments, removing them improved accuracy by 2%. Since they can be considered noisy data in binary classification, making the results with lower accuacry.

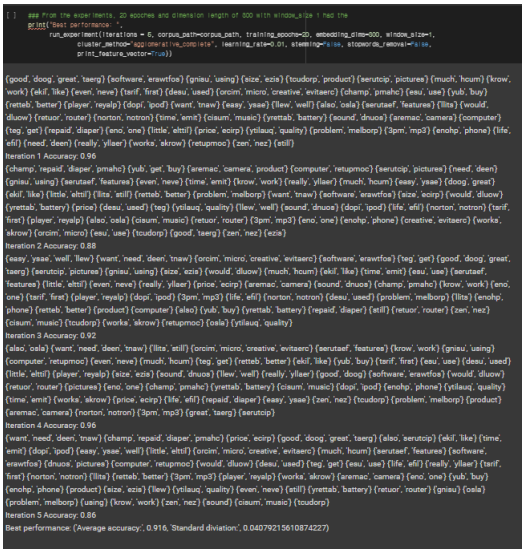
The best testing accuracy was obtained with an embedding size of 300 and a filter count of 100. A shortage of training data might explain why CNNs with more parameters did not outperform smaller ones. The more complicated models may have been overfitting for the minimal data supplied. The final hyperparameters used were 0.85 dropout rate, 300 embedding length, 100 CNN filters, mixed sentiment cleaning, and no stemming or word cleaning. The model is trained for 30 epochs and has an accuracy of 74%. With these hyperperameters, we gain from the improved generality provided by dropout, the absence of 'confusing' mixed samples, and more explicit embeddings.

---

[3] "Information Bottleneck for Multi-Task LSTMs - OpenReview." https://openreview.net/pdf?id=hnqkGVw_jtA. Accessed 19 Dec. 2022.

[4] "Convolutional Neural Network: Text Classification Model for Open ...." https://arxiv.org/pdf/1809.02479. Accessed 18 Dec. 2022.

Best performance from task 1:



Best performance from task 2: