Improvement on Word Representations using Sentiment Analysis

Abhijat Shrivastava
SBUID 112584928
ashrivastava@cs.stonybrook.edu

Aishwarya Vijayakumar
SBUID 112673842
hvijayakumar@cs.stonybrook.edu

Debapriya Mukherjee
SBUID 112683344
dmukherjee@cs.stonybrook.edu

December 13, 2019

Abstract

Word embeddings which consider the semantic and syntactic information from contexts are generally used for numerous natural language processing tasks. However the limitation is that they lack the ability to understand the underlying sentiment of the words. Analyzing the sentiment behind the words has become one of the most important research tasks in natural language processing currently. Sentiment analysis can be of huge importance and can be used in many applications such as advertising and marketing, business, politics and others. Various techniques are there for analyzing sentiments but lately, word embeddings method is widely being used in sentiment analyzing tasks. In this paper, we aim to improve the accuracy of the pre-trained word embeddings. We use a baseline model implemented with Word2Vec/GloVe methods and evaluate the accuracy of the word embeddings without Part-of-Speech (POS) tagging techniques and later with POS tagging techniques as our Idea 1, then evaluate the accuracy by implementing the bi-directional Long short-term memory (LSTM) model with POS tagging techniques. We evaluated the accuracy of the word-embeddings via different deep learning models and sentiment analyzing datasets. Our experimental evaluations show that the Improved Word Vectors (IWV) proposed in Idea 1, the implementation of Bi-LSTM model as proposed in Idea 2 are very effective for improving the accuracy of the word embeddings.

1 Introduction

The Word2Vec and GloVe require large corpuses for training and presenting an acceptable vector for each word. Investigators use pre-trained word vectors such as GloVe and Word2Vec because of the small size of some datasets. This would not be a best fit for the data. One more disadvantage of these word vector calculations is that they do not consider the context of a document. For example, the word vector for "beetle" as a car is equal to its word vector as an animal. These models ignore the sentiment information of the given text . The side effect of this problem is that words with opposite polarity are mapped into close vectors and it is a disaster for sentiment analysis.

Improved Word Vectors (IWV) is a method mentioned in the paper [1] which increases the accuracy of pre-trained word embeddings in sentiment analysis. This method is based on Part-of-Speech (POS) tagging techniques, lexicon-based approaches and Word2Vec/GloVe methods.

We demonstrate the advantage of specializing semantic spaces for either similarity or relatedness. Specializing for similarity is achieved by learning from both a corpus and a thesaurus, and for relatedness by learning from both a corpus and a collection of psychological association norms. We also compare the recently introduced technique of graph-based retrofitting (Faruqui et al., 2015) with a skip-gram retrofitting and a skip-gram

joint-learning approach.

All three methods yield specialized semantic spaces that capture human intuitions regarding similarity and relatedness significantly better than unspecialized spaces, in one case yielding state-of-the-art results for word similarity. More importantly, we show clear improvements in downstream tasks and applications: specialized similarity spaces improve synonym detection, while association spaces work better than both general-purpose and similarity specialized spaces for document classification.

2 Task

2.1 Baseline Model

A part of the baseline model being used for this task was available to us as a github implementation [3]. The dataset used for the training the baseline model is that of IMDB dataset.

We ran sentiment analysis on the IMDB dataset from the keras.datasets library for python. The model was trained using keras.models library for python.

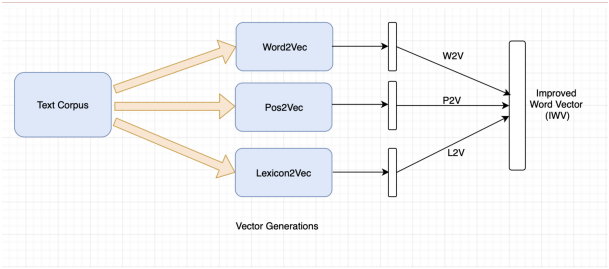

Figure 1: The main architecture of the the Improved Word Vector (IWV)

The baseline on which we worked is based on the paper "Improving the Accuracy of Pre-trained Word Embeddings for Sentiment Analysis" by Seyed Mahdi Rezaeinia et al [1]. The paper talks about 'Improved Word Vector' (IWV), which takes into account word2vec, pos2vec and lexicon2vec before producing the final embedding. We have evaluated the accuracies based on word2vec and pos2vec.

Word2Vec is a successful word embedding algorithm and is based on continuous Bag-of-Words (CBOW) and Skip-gram architectures which can provide high quality word embedding vectors. CBOW predicts a word given its context and Skip-gram can predict the context given a word. The generated vectors of words which appear in common contexts in the corpus are located close to each other in the vector space.

2.2 The issues

Word embedding techniques based on word2vec ignores the sentiment information of the given text . The side effect of this problem is that those words with opposite polarity are likely to be mapped into closer vectors and it might result in the loss of the true sentiments. [1]

The accuracy of the obtained model on a sentiment analysis task is 85%. This means that on a huge dataset, we'll lose a significant portion of the underlying true sentiments of the inputs. Our goal is to improve the word embeddings obtained from this technique using the three ideas mentioned below, so that true sentiments are retained while analysing a task.

3 Approach

3.1 Idea 1:

Part-of-speech (POS) tagging is an important and fundamental step in Natural Language Processing which is the process of assigning to each word of a text the proper POS tag.

Our idea 1 comprises of evaluating the improvement in accuracy of the word embeddings when computed with the Part-of-speech (POS) tagging techniques. The Part-of-speech gives large amount of information about a word and its neighbors, syntactic categories of words (nouns, verbs, adjectives, adverbs, etc.) and similarities and dissimilarities between them.

3.2 Idea 2:

For Idea 2, we thought of using a different method of generating the word vector embeddings instead of just using Word2Vec method. We used a Recurrent Neural Network (RNN) associated with Word2Vec network to achieve this. We applied Bidirectional LSTM in our model with the motive of improving the accuracy of the model.

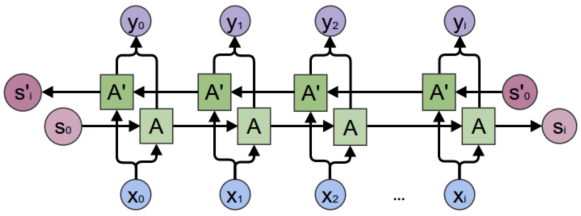

Figure 2: A Bi-directional LSTM model

3.3 Idea 3:

An idea that can be incorporated into our progress is that retrofitting of word vectors can be performed to further improve the training of the model. We target to minimize the euclidean distance between particular target vectors, so that the words with similar meaning get similar representations. This means that the words with same polarity will have their euclidean distance minimized. We propose this as a future scope of our project. The euclidean distance between two words can be represented by the formula:

$$\Psi(Q) = \sum_{i=1}^{n} \left[\alpha_i \|q_i - \hat{q}_i\|^2 + \sum_{(i,j)\in E} \beta_{ij} \|q_i - q_j\|^2 \right]$$

Figure 3: Representative objective function. alpha beta represents relative strengths of information from original word vector and neighbours

We observe that most embedding models depend on the correlation between the words appearing close to each other. The hypothesis behind this is that words occurring in similar contexts have similar meanings. But actually the words having similar contexts only means similar grammar, but not similar sentiment. So the vector spaces being trained from these models are suitable for analyzing word semantic relatedness(such as the relationship between keyboard and mouse), not word semantic similarity(such as the relationship between gasoline and diesel).

The idea 3 requires the usage of two thesauruses: the word similarity thesaurus from the electronic dictionary [MyThes] and the word relatedness thesaurus from the data set [USF Free Association Norms] to compare the similarity and the relatedness of the words. The paper[8] states the advantage of specializing semantic word embeddings for either similarity or relatedness. Here two variants of retrofitting and a joint-learning approach are taken care of, and capture human intuitions regarding similarity and relatedness.

3.4 Implementation Details

For implementing Idea 1, we used the baseline model and evaluated the model accuracy. We converted each generated POS tag to a constant vector and concatenated with Word2Vec vectors. The model was trained on 5 epochs, with a batch size of 1024. We found the accuracy to be 85%. The next step was to evaluate the model along with POS tagging technique to validate the significance of incorporating POS tags to the word embeddings.

For implementing Idea 2, we used a Bi-LSTM model associated with the Word2Vec model on the baseline model to generate the word embeddings.

4 Evaluation

4.1 Dataset Details

The datasets that have been used here [3] are as follows:

1. IMDB movie reviews dataset

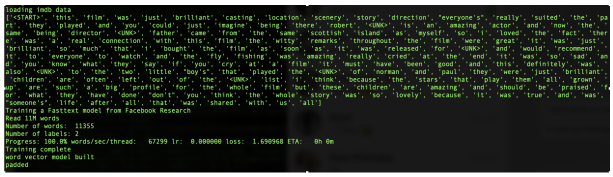

Figure 4: IMDB Dataset

2. Another IMDB movie review dataset [4]

   For evaluation of idea 2, we used a specific dataset which is well suited for sentiment analysis. A sentiment is either positive or negative. It has been categorized into 2 classes: 0 (for IMDB rating <5, and 1 (for IMDB rating ≥ 7. Each movie has an upper bound of 30 reviews. The test and training dataset are different from each other. We have used the following files as our dataset:

   (a) labeledTrainData - The labeled training set. The file is tab-delimited and has a header row followed by 25,000 rows containing an id, sentiment, and text for each review.

   (b) testData - The test set. The tab-delimited file has a header row followed by 25,000 rows containing an id and text for each review. Your task is to predict the sentiment for each one.

   (c) unlabeledTrainData - An extra training set with no labels. The tab-delimited file has a header row followed by 50,000 rows containing an id and text for each review.

3. Two thesaurus used for checking the similarity and relatedness between words for Idea 3.

4.2 Evaluation Measures

Idea 1 was evaluated on dataset 1. Idea 2 was implemented and validated against dataset 2.

The metrics for the above mentioned ideas are: accuracy, loss, precision, recall and f1-loss.

For idea 3, the metrics are the learning rate and the loss of the model and pearson co-efficient while calculating the similarity and the relatedness of the word vectors.

4.3 Baselines

For ideas 1 & 2:

A part of the baseline model was available to us. We made some changes to the available baseline in order for it to work on our systems. Certain libraries required by the baseline were installed on our conda environment.

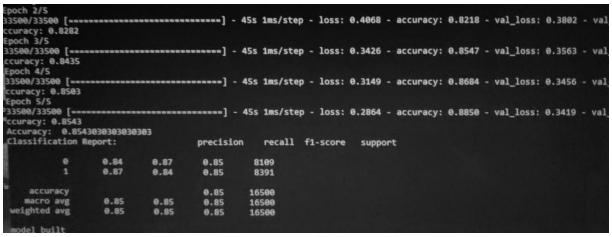

Figure 5: Result after running the baseline model

For idea 3:

We obtain a part of another baseline model for the paper mentioned in [8] in github [9] to check the similarity and relatedness between word embeddings based on two available thesaurus [7,8]. We made certain changes in the baseline to make the model run on our systems and many requirements needed to be installed for that.

We executed the baseline model for this on our systems and observed the following:

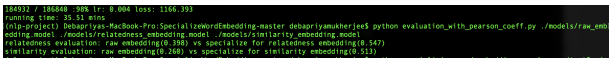

Figure 6: Executing the baseline model for the Idea 3

1. relatedness evaluation: raw embedding(0.398) vs specialize for relatedness embedding(0.547)

2. similarity evaluation: raw embedding(0.260) vs specialize for similarity embedding(0.513)

We have an executable baseline for this and we can extend the idea behind the baseline as a part of our future scope to work on for improving the word embeddings.

4.4 Results

Idea 1:

On comparing the accuracies of the word embeddings with and without POS tagging technique, we see an accuracy improvement of 1% after using POS tagging on the words.

Model (Evaluated in 5 epochs)	Accuracy
With POS tags	86.0%
Without POS tags	85.0%

Figure 7: Result for evaluating word embeddings with and without POS tagging technique

Idea 2:

In this idea, we have implemented a Bi-LSTM model to generate the word embeddings. On implementing the Bi-LSTM model, we see a significant increase in the accuracy of the model. We evaluate it on 20 epochs and observe the following over the 20 epochs:

1. We train the model on 23750 samples, and validate on 1250 samples.

2. The accuracy gradually increases from 0.6306 to 0.9805.

3. The loss gradually decreases, starting from 0.6439 at the first epoch and ending with 0.0547 at the last epoch.

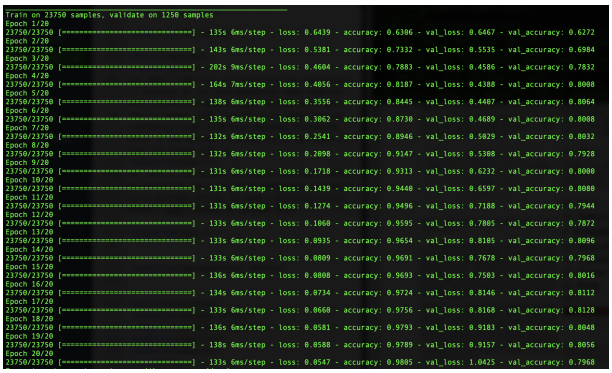

Figure 8: Evaluating the accuracies in 20 epochs on implementing Bi-LSTM model

The Bi-LSTM model is a bidirectional recurrent neural networks for document-level sentiment analysis with word2vec Embedding. The model was implemented with Bi-LSTM model, and the experiment shows that the Bi-LSTM model associated with word2vec word embedding outperformed the baseline model and achieved a 98.05% accuracy.

4.5 Analysis

Idea 1:

We analyzed our results by comparing the accuracy of the models on a sentiment analysis task on a IMDB dataset obtained from keras.datasets library, available for python. We evaluate the system once without using the POS tagging techniques and then using it. On comparing both, the results show that the system appears to work well when Part-of-Speech tags are incorporated into the word embeddings, since the accuracy increases by 1% (for 5 epochs). This shows that incorporating POS tags into word embeddings helps them absorb more sentiment-related information, which can be put to use in sentiment analysis tasks.

Idea 2:

On evaluating the accuracies of the word embeddings with POS tagging techniques, we found that there is a slight improvement. On implementing the Bi-LSTM

model associated with the Word2Vec model along with the POS tagging technique (idea 1) to generate the word embeddings, we observed a significant improvement in the word vector representations. This improvement is supported by the boost in accuracy of the sentiment analysis task.

4.6 Code

The project we've been working on has been uploaded on the following Github repository: https://github.com/abhijat96/NLPProject

5 Conclusions

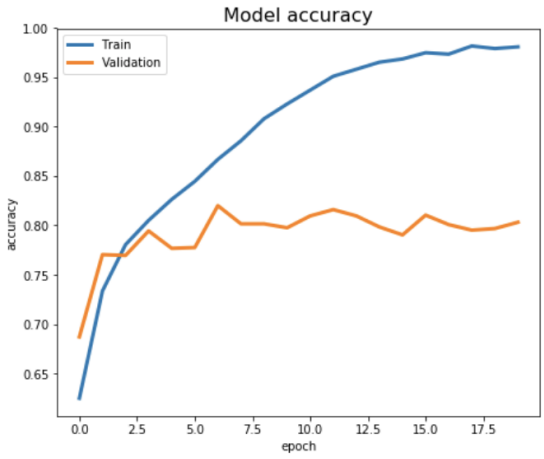

Figure 9: Graph of accuracy of model

From the graph above, we observed that the model accuracy keeps increasing gradually as the number of epochs increases on implementing the biLSTM model associated with the word2Vec model, incorporated with POS tagging techniques. Thus we can conclude that the final model shows a significant improvement on the baseline model. Therefore, we have improved the accuracy of the pre-trained word embeddings.

References

[1] Seyed Mahdi Rezaeinia, Ali Ghodsi, Rouhollah Rahmani.
Improving the Accuracy of Pre-trained Word Embeddings for Sentiment Analysis.

[2] Liang-Chih Yu, Jin Wang, K. Robert Lai and Xuejie Zhang
Refining Word Embeddings for Sentiment Analysis.

[3] Github Link (Baseline 1):
https://github.com/PrashantRanjan09/Improved-Word-Embeddings.

[4] Kaggle Link for Data:
https://www.kaggle.com/jiaofenx/imdb-review-word2vec-rnn-tutorial/data.

[5] Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, Noah A. Smith
Retrofitting Word Vectors to Semantic Lexicons.

[6] Thesaurus:
http://www.openoffice.org/lingucomponent/thesaurus.html.

[7] Thesaurus:
http://w3.usf.edu/FreeAssociation/.

[8] Douwe Kiela, Felix Hill and Stephen Clark
Specializing Word Embeddings for Similarity or Relatedness.

[9] Github Link (Baseline for Idea 3):
https://github.com/huangmiaoxin/SpecializeWordEmbedding.