Paragraph Vector

Tianyi Sun[*]and Qingyuan Xue[†]

May 28, 2022

1 Introduction

We implemented the neural net model "Doc-To-Vector" [Le and MikolovLe and Mikolov2014], which generates the sentence-level feature embedding, and conducted the benchmark analyses including semantic textual similarity and corpus of linguistic acceptability. The goal of the project is to investigate that whether and to what extend "doc-To-vector" has the capacity to capture the syntactic and semantic meaning of input sentences, by using the generated embedding for downstream linguistic tasks. Although the computational implementation is relatively easy through API, the challenge is likely to be found in benchmark analysis, where there is no absolute measure to test the linguistic capacity of the model. In fact, performance of the model depends on many factors, which can be related to the hyper-parameter setting. In addition, the task itself of generating the sentence-level numerical representations are challenging, since the success of the task depends on the design of linguistic tasks under training. The challenges mentioned above are important as a well-designed language model pre-trained on millions of corpus are expected to yield human-comparable performance on linguistic tasks, which in turn will facilitate our understanding of how the model process and understand the language. Moreover, defining and testing the appropriate linguistic benchmark are the keys for us to assess the strength and weakness of the model.

2 Background

Finding the textual vector representations for machine to process the language is fundamental yet challenging task in applications of natural language processing, due to the ambiguity presented in all levels of language [Sezerer and TekirSezerer and Tekir2021]. In analogy with human's processing of texts, semantic clues as well as the context and world knowledge are all required, and hence two different computational systems are proposed for language processing. One is the knowledge system, the other is the statistical system for vector representations.

Among the methods of statistical system, one-hot encoding of binary vector space is a simple technique. In each word vector, one-hot encoding uses the positional indices one to indicate the corresponding word, and zero elsewhere in the vector, which result in a three dimensional tensor representing the doc or paragraphs. For a documents including N unique words, the last dimension of the tensor should be N. However, the drawback with this technique is that for very large corpus and hence large N, the vector representation will be very sparse, which will lead to difficulties in computational processing. Moreover, one-hot encoding does capture the contextual and semantic information such that similarity of two different words can not be compared.

Another similar simple statistical model is the bag of words representation which focuses on multiplicity, which, however, disregard the order of the words and account for little semantic information. The word vectors [Mikolov, Chen, Corrado, and DeanMikolov et al.2013] addressed the issue present in one-hot encoding. The neural network model [Mikolov, Chen, Corrado, and DeanMikolov et al.2013] can learn high quality word vectors with relatively low computational costs. It is also found that learned vectors are able to capture the syntactic and semantic similarities. Inspired by the same idea of utilizing the unsupervised

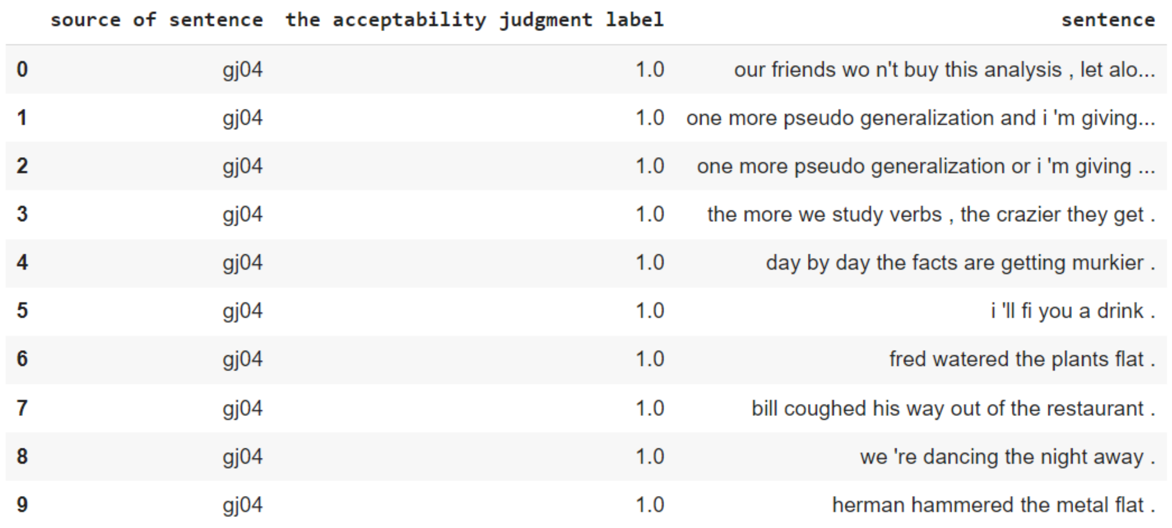
[*]The University of Chicago
[†]The University of Sydney

training, the method of paragraph vector is proposed to generate sentence level embedding. The method, in the next word prediction task, also uses the unique paragraph token as the feature input. The paragraph vector is able to outperform all other state-of-the-art models such the word of vector averaging on classification and sentiment tasks. However, according to the result of our analysis, the model is yet to sufficiently capture the knowledge of syntactics.

3 Data

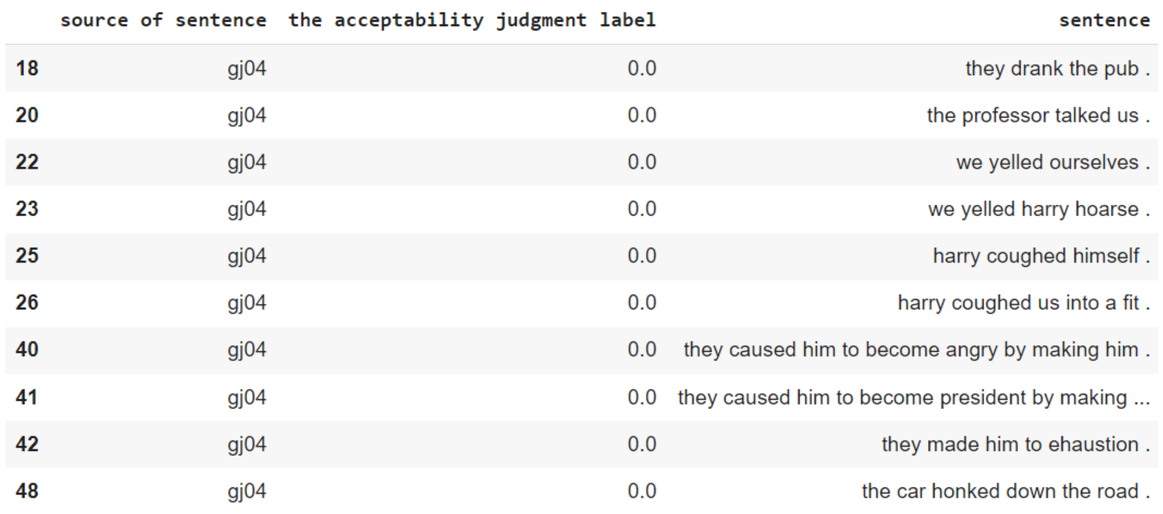
3.1 Neural Network linguistic acceptability Judgments

Following the work of linguistic acceptability Judgments by Alex Warstadt (2018), we uses the data set of acceptability-corpus provided, which has 8550 in field examples of training and 526 examples of testing. Each example is a phrase labeled by the acceptability judgment score, which uses 1 to denote the sentence with no grammar mistakes and 0 otherwise. The following are the 11 positive examples from the training set:



	source of sentence	the acceptability judgment label	sentence
0	gj04	1.0	our friends wo n't buy this analysis , let alo...
1	gj04	1.0	one more pseudo generalization and i 'm giving...
2	gj04	1.0	one more pseudo generalization or i 'm giving ...
3	gj04	1.0	the more we study verbs , the crazier they get .
4	gj04	1.0	day by day the facts are getting murkier .
5	gj04	1.0	i 'll fi you a drink .
6	gj04	1.0	fred watered the plants flat .
7	gj04	1.0	bill coughed his way out of the restaurant .
8	gj04	1.0	we 're dancing the night away .
9	gj04	1.0	herman hammered the metal flat .

(a) For complete illustration, here are the 11 positive examples.



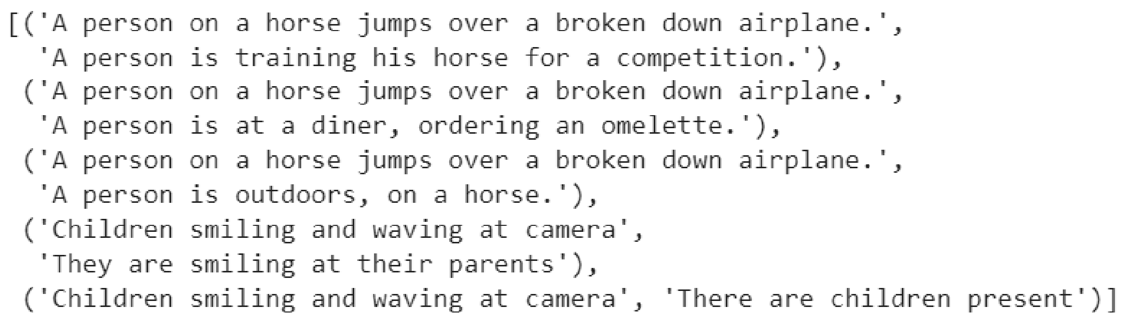
	source of sentence	the acceptability judgment label	sentence
18	gj04	0.0	they drank the pub .
20	gj04	0.0	the professor talked us .
22	gj04	0.0	we yelled ourselves .
23	gj04	0.0	we yelled harry hoarse .
25	gj04	0.0	harry coughed himself .
26	gj04	0.0	harry coughed us into a fit .
40	gj04	0.0	they caused him to become angry by making him .
41	gj04	0.0	they caused him to become president by making ...
42	gj04	0.0	they made him to ehaustion .
48	gj04	0.0	the car honked down the road .

(b) For complete illustration, here are the 11 negative examples.

3.2 Sentence Similarity test

To test the document to vector's ability to capture the semantics. We use the data set of Stanford Natural Language Inference (SNLI) Corpus of 570152 pairs of premise and hypothesis, which are labeled by entailment, contradiction, or neutral to indicate the relationship in each pair[1] Below are the examples of the five pairs. The data set can be used to measure evaluate the embedding similarity given the label for each pair.

3.3 IBDM movie review



```
[('A person on a horse jumps over a broken down airplane.',
  'A person is training his horse for a competition.'),
 ('A person on a horse jumps over a broken down airplane.',
  'A person is at a diner, ordering an omelette.'),
 ('A person on a horse jumps over a broken down airplane.',
  'A person is outdoors, on a horse.'),
 ('Children smiling and waving at camera',
  'They are smiling at their parents'),
 ('Children smiling and waving at camera', 'There are children present')]
```

Figure 2: IBDM movie review dataset

Figure 1a and 1b, the data set has 50k labeled movie reviews and 50k unlabeled movie reviews. The sentiment labels have either 1 for positives or 0 for negatives. Each movie review is treated as one long paragraph.

4 Code and algorithms

The major algorithm we focus on is the document-to-vector model, provided by the GENSIM interface. The model uses the paragraph unique token D and other word tokens present in the model as inputs, map the inputs to the vector space through neural nets, concatenate or average the word vectors, and pass the vector through the classifier for the next word prediction. The training phrase is through backward propagation, where parameter weights in the model are updated each time using the gradient. For inference, the model weights are freezed; given the input of a new document, the model output its embedding. However, for specific document, the prepossessing is required.

The first step is to split the text of strings into separate tokens. The second step is to normalize the tokens such that the unwanted characters are removed and capital letters are lowered. The third step is to tokenize each list of tokens such that the common stop words with little semantics are removed. All the codes [2] are provided by Susan Li. There are two major algorithms for the benchmark learning tasks we have tested the language model of doc-to-vector. The first is the logistic regression classifier, which we have implemented from the ski-learn python library. This regressor is applied for the Neural Network linguistic acceptability Judgments and IMDB movie review sentiment classification task.

The second is the neural net based logistic classifier for the task of Neural Network linguistic acceptability Judgments. The classifier has one hidden layer and Sigmoid activation function in the output layer. The hidden units is set to be 50.

More importantly, for the parameter settings of document-to-vector model, there are some important parameters we have set up through GENSIM library. The dimensional size of each vector corresponding to each paragraph is set to be 400,following the experimental setting of the paper; number of negative words is two, initial learning rate is 0.001, window-size is 10. Specifically, two training models are implemented simultaneously, one of which is using the distributed memory and another using the bag words model.

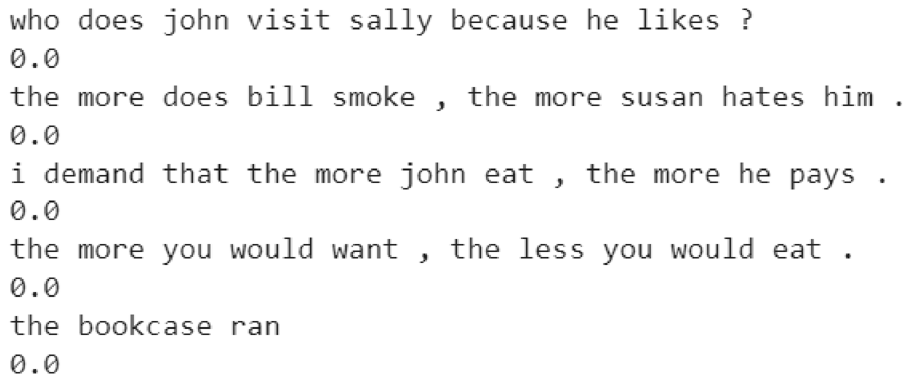
[1] https://nlp.stanford.edu/projects/snli/

[2] https://towardsdatascience.com/multi-class-text-classification-with-doc2vec-logistic-regression-9da9947b43f4

The two vectors from the two models are then concatenated as inputs for the classifier. For the combined model implementation, we first train each model for 15 epochs, concatenate two models into one, learn the vectors from the training docs and testing docs, train the classifier, and get the results. For implementation, Doc2Vec model is initialized and trained through GENSIM library.

5 Results and Analysis

We tested the doc-to-vector model on the data set of linguistic acceptability Judgments, and have found that the model exhibits poor performance of 69 percent accuracy, similar to the random guessing rate and comparing to the average accuracy of 86.1 percentage from human annotator. All other embedding models including BNC, GloVe, and ELMo-style have achieved at least 70 percentage of accuracy on the task. Based on the testing result, we may conclude that the doc embedding capture very little syntactic information. However, it should be noted that further detailed analysis are needed for different sentences of different source fields. For illustration, we look at few following examples where the model has wrong acceptability judgment.



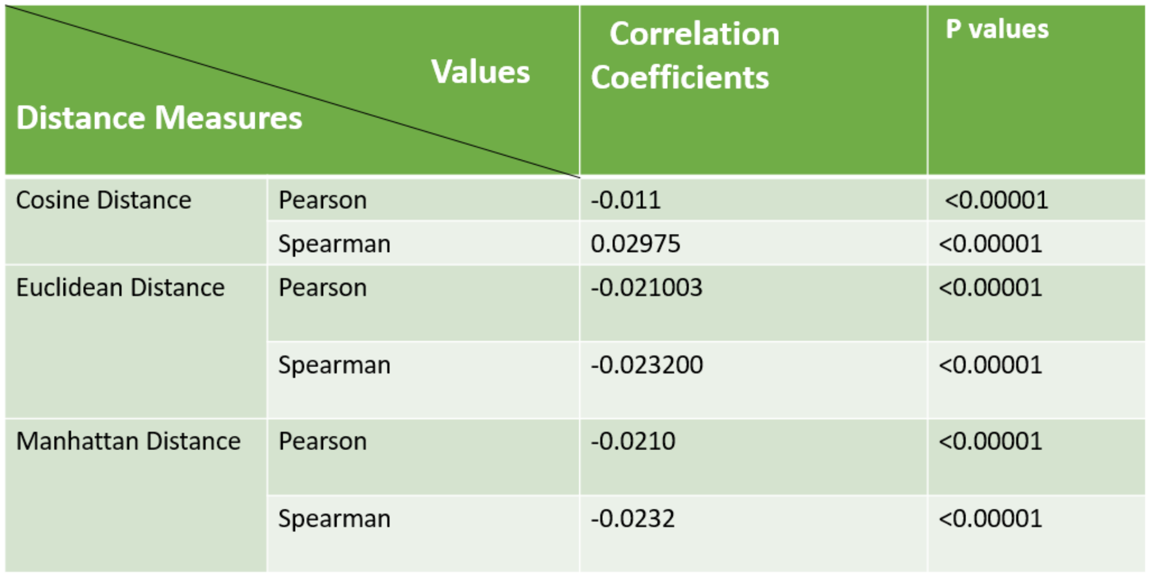
```
who does john visit sally because he likes ?
0.0
the more does bill smoke , the more susan hates him .
0.0
i demand that the more john eat , the more he pays .
0.0
the more you would want , the less you would eat .
0.0
the bookcase ran
0.0
```

Figure 3: This shows the false positive from the model, that is for each sentence with grammar errors, the model failed to identify them and label the sentence as acceptable.

We have also conducted the sentence similarity test using the Stanford Natural Language Inference (SNLI) Corpus. we have no full results ready yet, but we are planing that for each pair in SNLI, we will calculate the cosin-distance scores, manhattan-distances, and euclidean-distances. For each of the three distance metrics, we will be then deriving the Pearson correlation scores, with respect to the label. Based on the result, we can then access the capacity of the model to capture semantics.

On the smaller data set for sentiment analysis during COVID-19, however, we have tested the corpus embedding from the mode and the sentiment label to fit a logistic model. The results are also relatively poor. The testing accuracy is only 32 percentage and F1 score of 34 percentage. The results are well-below the random guessing accuracy rate. An appropriate learning task is found as the movie review sentiment prediction task, with the data containing 25000 labeled training set, 25000 labeled test sets, and 50000 unlabeled reviews. Following the paper, we train the language model on the 25000 labeled and 50000 unlabeled reviews, learn the vectors for the training set, train the logistic classifier with the learned vectors, learn the vectors for the testing test, and predict the label using the trained logistic classifier. Although we are not able to replicate the result in the paper, we have indeed obtained the accuracy of 75 percent, which is well-above the chance rate of 50 percent. Without the original code producing the results from the paper, we hence can not identify the reason for the failure.

Another learning task we have tested the language model on is the sentiment similarity task. The results are reported in the following table 4.



Values / Distance Measures		Correlation Coefficients	P values
Cosine Distance	Pearson	-0.011	<0.00001
	Spearman	0.02975	<0.00001
Euclidean Distance	Pearson	-0.021003	<0.00001
	Spearman	-0.023200	<0.00001
Manhattan Distance	Pearson	-0.0210	<0.00001
	Spearman	-0.0232	<0.00001

Figure 4: We have obtained the following results, which however indicates the model's weak capacity in capturing the semantic information. For future work, it is hence expected that we will focus on to investigate the reasons, while we have excluded the possibility of coding errors.

References

[Le and MikolovLe and Mikolov2014] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*. PMLR, 1188–1196.

[Mikolov, Chen, Corrado, and DeanMikolov et al.2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. arXiv:cs.CL/1301.3781

[Sezerer and TekirSezerer and Tekir2021] Erhan Sezerer and Selma Tekir. 2021. A Survey On Neural Word Embeddings. arXiv:cs.CL/2110.01804