Fayoum University Journal of Engineering, 2022, Vol: 5(1), 66-73
https://fuje.journals.ekb.eg/
ISSN Online: 2537-0634
ISSN Print: 2537-0626

Study of different Statistical Machine Learning Techniques for Text Sentiment Classification

Abdelrahman N. Taha[1,*] & Rania Ahmed Abdel Azeem Abul Seoud[2]

[1]Electrical Engineering Department, Faculty of Engineering, Fayoum University, Fayoum 63514, Egypt
[2]Professor, Electrical Engineering Department, Faculty of Engineering, Fayoum University, Fayoum 63514, Egypt

*Corresponding author: Abdelrahman N. Taha (ant11@fayoum.edu.eg).

How to cite this paper: Taha A.N. & Abul Seoud, R.A.A. (2021). Study of different Statistical Machine Learning Techniques for Text Sentiment Classification. *Vol. 5 (1),* pp. 66-73.

Abstract

Text classification is an important task in NLP for various applications from movie review classification to market analysis. NLP as a tool provides the capability to process huge amount of text and come up with conclusions. In this paper we investigate statistical machine learning for NLP for document classification. The target problem of choice is sentiment analysis, we explore various techniques for text preprocessing, feature selection and model selection to find a good fit model. This paper acts as both a system proposal and also a primer for those who to start practicing NLP, we try to provide insight and intuition about modelling choices for text classification that extend even beyond the task scope to general NLP. In this paper we propose a feature based text sentiment analysis relying heavily of the BoN (Bag of N-grams) model and utilizing these features with a statistical ML classifier. We use the IMDB movie review dataset (Maas et al. 2011) for benchmarking.

Keywords

Machine Learning; Sentiment Analysis; Natural Language Processing; IMDB Sentiment Analysis; Text Classification.

1. Introduction

Text sentiment analysis is the classification of the sentiment of the author of a text document from a set of predefined classes. Sentiment analysis in essence is a classification problem defined as, given the text document provide the class label representing the sentiment.

The task at hand involves categorizing movie reviews according to author sentiments as belonging to one of two possible classes $\{\mathrm{POS}, \mathrm{NEG}\}$. This constitutes a binary classification task relative to the text documents representing the movie reviews. Our interest in this work revolves around supervised classification. This can be formulated as follows, given a dataset $D = \{(x, y): x \in X, y \in Y\}$ where X is the set of all features corresponding to the document and Y is the set of all labels in the training set. Find the model that maximizes the likelihood of the prediction. For supervised learning this is formulated as in Equation (1)

$$\hat{y} = \mathrm{argmax}(\mathrm{P}(\mathrm{y}|\mathrm{x})), \qquad (1)$$

A lot of approaches already exist for text document classification, the main approaches can be categorized into two major approaches feature based text classification and sequence modelling based text classification.

In feature based approaches, the text is not treated as a sequence of words with temporal relations, but as an unstructured data source to be used for feature extraction, said features are then processed by using dimensionality reduction techniques such as LSA (Latent Sematic Analysis), PCA (Principal Component Analysis) or by using statistical based feature selection such as selection via information gain (IG) or using a statistical distribution (e.g. the chi-squared χ^2 distribution). The selected features are then fed to a classification algorithm to perform the task against the labels (supervised classification) or by exploiting similarities (unsupervised classification). An example for a feature based approach is using the bag of words model of language to represent a document by its token count. Then use a simple classifier such as Naïve Bayes classifier to perform binary classification.

The other approach is to try to model the temporal relation between words in a sentence and represent that knowledge such that it captures word sequence relations. This approach is the predominant approach today in NLP especially for complex problems such as language modelling and machine translation. An example of this is using a deep neural RNN network such as an LSTM as proposed in (Hochreiter & Schmidhuber 1997) or by leveraging more complex model such as transformers (Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez et al. 2017) to learn a representation from training data by using unsupervised learning techniques. Much like MLM (masked language modelling) used in BERT (Devlin, Chang, Lee & Toutanova 2018).

The main focus in this work is based on the feature based approach to show it performance compared to some other state of the art approaches. Our approach is to use multiple machine learning classifiers trained on bag of n-gram features. We can divide our approach into three main efforts:

- Preprocessing
- Feature Extraction
- Model Selection

The main focus is to prove that this system will provide adequate performance given its relative simplicity. Some preprocessing techniques and feature selection used with bag of n-grams to increase model performance are tested to find the best combination for a good system. The dataset used in this work is the IMDB movie review dataset (Maas, Daly, Pham, Huang, Ng, & Potts 2011). The overall system architecture is illustrated in figure (1).

The rest of the paper is organized as follows, section 2 discusses related work, section 3 discusses the methods used for this study, section 4 the experiments conducted and the results obtained also shows our best result against SOTA (State of the Art) results, the conclusion is in section 5 and future work in section 6.

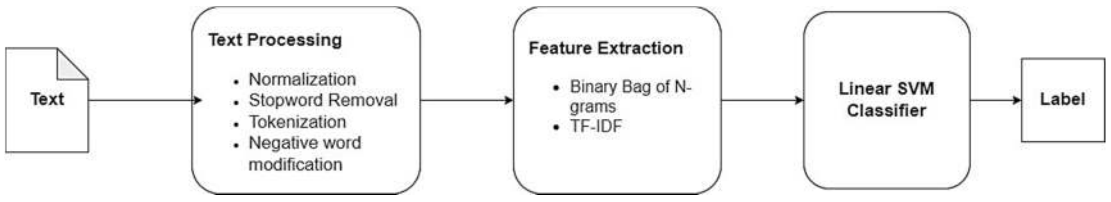

Figure 1. Illustration of the proposed model (Best model).

2. Related Work

In this section, the related work on text classification and sentiment analysis is shown for machine learning based approaches. These approaches can be categorized broadly as feature based approach and a sequence modelling approach. Other techniques exist such as using sentiment orientation and heuristic based (prior knowledge) approaches. For example (Turney 2002) used sematic orientation of phrases with two preselected words “excellent” and “poor” to classify reviews as recommended or not. However the focus in this section is on the two aforementioned approaches.

2.1 Feature Based Approach

This approach disregards the composition of documents and the relation between words in sentences and treats each word as an independent unit in terms of order. Systems built that way are simple and quite effective for text classification. Multiple works discuss these systems and show good results.

(Joachims 1998) proposed a system based around an SVM classifier that trains on bag of word features with information gain (IG) criterion to select only a subset of features. He reported that SVM (Support Vector Machine) outperformed some other ML techniques such as KNN (K-Nearest Neighbors) and Naïve Bayes.

In (Pang, Lee & Vaithyanathan 2002) the authors use three ML classifiers Naïve Bayes, maximum entropy and SVM for movie review sentiment classification, the authors compared different approaches for feature extraction such as using term frequency vs term presence and adding bigrams and POS (part of speech) tags as features. SVM trained on unigram presence (binary) features were reported to perform best.

(Wang & Manning 2012) build on top of previous work on Naïve Bayes and use MNB (Multinomial Naïve Bayes) classifier, SVM and their proposed mixture of the two NBSVM where they use Naïve Bayes log features as input to train an SVM classifier.

Other approaches don’t necessarily use sparse bag of words/n-grams representations, but use instead dense representation (embeddings) for words or sentences for training more complex classifiers (such as neural nets) leveraging the lower dimensionality of this representation. Some popular word embeddings were introduced in word2vec (Mikolov, Chen, Corrado & Dean 2013) and Glove (Pennington, Socher, & Manning 2014). These embeddings are useful for the feature based approach and also for training sequential models. Word embeddings can be averaged to generate sentence embeddings for text classification.

In (Le & Mikolov 2014) introduce paragraph vector PV which is a similar concept to word embeddings but used for paragraphs. PV is trained in two ways, PV-DM (Paragraph Vector – Distributed Memory) where the document embedding matrix is trained to predict the next word using a classifier trained on PV averaged/concatenated with the word embeddings of the previous words. The other PV method is PV-DBOW (Paragraph Vector – Distributed Bag of Words) here the PV is trained using a classifier on a random sample from a random text window without using any context. PV features can be used as input to ML classifier for text classification.

DV-ngram proposed by (Li, Liu, Du, Zhang & Zhao 2015) builds on the PV idea by predicting n-grams in paragraphs in addition to predicting words. Weighted neural bag of n-grams introduced in (Li, Zhao, Liu, Wang & Du 2016) build on concept of DV-ngram by using Naïve Bayes weights for the words (to mark their significance) when training the text embedding vectors.

In (Thongtan & Phienthrakul 2019) the authors utilize cosine similarity instead of regular dot product similarity for training document embedding vectors, their work combined with neural bag of n-grams with NB weights produces a very formidable accuracy score of 97.42%.

2.2 Sequence Based Approach

This approach models the context of the text and relation between the words in a sentence. Multiple methods are used to model context in text, some utilize recurrent models such as RNNs specially LSTMs (Long short term memory) neural networks.

In (Zhang, Liu, and Song 2018) build on top of the idea of a Bi-LSTM (Bidirectional LSTM) by creating a sentence level LSTM network that contains hidden states for each word in the text sequence in parallel and contains a sentence level state to capture sentence representation in addition to text context representation. They demonstrate the S-LSTM (Sentence LSTM) performance on text classification and show gains compared to regular Bi-LSTM.

Another method is to use representation learned by a recurrent model for classification similar to work in (Peters, Neumann, Iyyer, Gardner, Clark, Lee, et al. 2018) where representations from a Bi-LSTM language model are used as features for multiple downstream tasks including text classification. The system dubbed ELMo (Embeddings from Language Models) utilizes the representations at various layers in the LM to represent tokens.

Other approaches based on the success of BERT (Devlin, Chang, Lee & Toutanova 2018) and transformer based representation learning from text utilize representations learned from complex transformer models to be used as features for downstream tasks.

(Sun, Qiu, Xu, & Huang 2019) illustrate how to fine-tune a pre-trained BERT model with different strategies and their effect on performance on downstream tasks. They use strategies such as single task fine-tuning, multi-task fine-tuning and continuing pre-training using in-domain/in-task data.

In this work the focus is mainly on the traditional bag of words/n-grams models with ML classifiers such as Naïve Bayes and SVM and investigate the effect of various preprocessing and feature selection techniques on performance.

3. Methods

The methods used revolve around testing various machine learning models on features extracted from bag of n-gram sparse representation of the input text. Our pipeline can be divided into three major steps:

- Preprocessing: where the text is prepared for feature extraction
- Feature extraction: using a bag of n-gram representation with feature selection to represent documents.
- Classification: using the representation from feature extraction to train a binary classifier.

The final system is built incrementally where the best result from each step is used as a base for the next step and so on.

3.1. Preprocessing

The following methods for text preprocessing are used

- Tokenization
- Normalization
- Stop-word removal
- Stemming
- Lemmatization
- Negation handling

Tokenization is the way we split the text input into separate tokens (words for example). Tokenization used splits around white space between sequences of valid characters. Tokenization is done by default in all of our experiments including the baseline.

Text normalization is transforming all text to lower case and removing special symbols and punctuation marks.

Stop-words are frequent words such as (and, or, is …etc.) that have no significant effect on the overall sentiment of the document when using bag of n-gram approaches.

Stemming is the process of removing suffixes from words to capture meaning instead of form.

Lemmatization is a more in depth form of stemming where each word is reduced to its base root “lemma” using morphological analysis. WordNet Corpus (Miller 1995) based lemmatizer was used for our experiments.

Negative words such as “didn’t like” are handled by introducing a tag NOT_ before words after negation (“didn’t like” becomes “didn’t NOT_like”). This method illustrated in (Jurafsky & Martin 2022) helps feature based models to capture negation in the absence of context.

3.2 Feature extraction and selection

Features are extracted using a bag of n-gram model where each sentence is represented as a sparse vector $x_i \in R^N$ for sentence i. Where N is the number of all possible n-grams. Our approaches for feature extraction:

- Different orders of n-grams
- Using TF (Term Frequency Features)
- Using IDF (Inverse Document Frequency)
- Using Binary “presence” features

TF is usually used with IDF or alone (TF/TF-IDF) the two steps are performed after extracting n-gram counts

- Term Frequency
  The word counts are normalized relative to the document they exist in, the intuition here is to balance out the effect of long and short documents by normalizing both features
  $$TF = \frac{word\ count}{total\ words\ in\ document}, \quad (2)$$
- Inverse Document Frequency
  Calculated as the log of the frequency of word occurrence in other documents across the data set, the intuition here is to limit the effect of frequent words from having too much representation

Binary features are used to simply indicate if an n-gram exists in a sentence or not. They can be combined with TF-IDF.

Features selected using bag of n-grams are sparse with very high dimensionality. They can be further reduced without significant loss in model performance by using feature selection such as information gain or using a probability distribution such as χ^2. Two methods for feature selection were used, using the maximum 1024 frequent features (based on the entire dataset) and the most probable 1024 features according to a χ^2 distribution.

3.3 Classification

Statistical ML contains a plethora of classifiers, each performing classification in a different way. The following ML classifiers were trained and tested on the IMDB dataset.

- Linear SVM
- SVM trained with SGD
- Ridge Classifier
- Perceptron Network
- K nearest neighbors
- Random Forest
- Bernoulli Naïve Bayes
- Complement Naïve Bayes

4. Experiments & Results

4.1 Experiment setup

All experiments were conducted on Google Colaboratory for computation and quick testing. NLTK (Natural Language Toolkit) (Bird, Klein & Loper 2009) was used for lemmatization and stop-word removal. Sci-kit Learn (Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, et al. 2011) python library was used for model training, feature extraction and parameter tuning. The training set of the IMDB dataset is used with 25K training examples split evenly between negative and positive examples. All scores are produced by testing against the test split provided by the dataset of 25K test examples also evenly split between positive and negative examples.

4.2 Results

Metrics used are accuracy, recall, precision and F1-score, with F1-score being used as the deciding factor for the best approach when selecting.

The first result is our baseline model based on a Multinomial Naïve Bayes classifier with unigram features (bag of words) it achieves accuracy of about 81.5% (see table (1) for detailed metrics).

Preprocessing experiment results are shown in table (1) the classifier used here is still a multinomial Naïve Bayes classifier. From the first row text normalization achieves marginal gain, adding lemmatization to the normalization seems to degrade performance. Adding stemming instead of lemmatization with normalization degrades performance more (indicating that lemmatization is better). However both lemmatization and stemming didn't improve performance.

Table 1. Preprocessing Experiment Results (best results in bold).

Preprocessing Results				
	accuracy	**Recall**	**Precision**	**F1 Score**
Baseline	81.5%	75%	86.2%	80.2%
Method				
Text Normalization	81.6%	75.4%	86.2%	80.4%
Normalization+ Lemmatization	81.4%	75%	85.9%	80.1%
Normalization + Stemming	81.1%	74.7%	85.6%	79.8%
Normalization + Stop word removal	82.2%	76.6%	86.2%	81.1%
Normalization + Stop word removal + Negative word modification	**82.4%**	**76.7%**	**86.6%**	**81.4%**

Removing stop-words generated a good gain of more than 1% in accuracy and F1-score. Finally combining normalization with stop-word removal and negation handling yields the best pre-processing result on our basic classifier.

Feature extraction & selection results are shown in table (2). Building on top of the best result from pre-processing experiments and using the same base classifier various feature extraction and selection methods are tested. First using Term Frequency only on word counts yields more than 2% absolute gain. Multiple n-gram orders were tried the best result obtained was at 4-gram features and provides a 3.5% absolute improvement over regular unigram features.

Table 2. Feature Extraction & Selection Results.

. Feature Extraction & Selection Results				
	accuracy	**Recall**	**Precision**	**F1 Score**
Previous Best	82.4%	76.7%	86.6%	81.4%
Method				
Word count + TF	84.5%	80.6%	87.4%	83.8%
Up to 4-grams features	85.9%	82.5%	88.6%	85.4%
4-grams + binary counts	86.6%	82.9%	89.5%	86.1%
4-grams + binary counts + TFIDF	**87.3%**	84.1%	**89.9%**	**86.9%**
4-grams + binary counts + TFIDF (top 1024)	83.3%	84.4%	82.6%	83.5%
4-grams + binary counts + TFIDF (χ^2 top 1024)	86.4%	**88.7%**	84.8%	86.7%

Building on top of the 4-gram result and using binary "presence" features a further 0.7% gain in performance is observed. Using TF-IDF with the settings in row 3 of table (2) provides an additional gain of about 0.7% in accuracy. However using the most frequent 1024 features tends to degrade performance considerably. Using χ^2 feature selection also degrades performance but not as much as regular most frequent selection. A worthwhile note here is that χ^2 selection provides the best recall for feature extraction, but degrades precision considerably. Row 4 in table (2) result (4-gram + binary features + TF-IDF) are selected as best based on F1-score for the next stage of experiments.

The next stage is classifier model testing, the models specified in section 3.3 are tested with SVM classifiers trained once with L1 Loss and once with L2 loss. We find from the results obtained in table (3) that the best classifier mode for our case was the Linear SVM model trained with L2 penalty. This confirms similar conclusions drawn in (Joachims 1998) and (Pang, Lee & Vaithyanathan 2002).

Table 3. Classifier Selection Results.

Classifier Selection Results				
	accuracy	**Recall**	**Precision**	**F1 Score**
Previous Best	87.3%	84.1%	89.9%	86.9%
Method				
Linear SVM L2 Penalty	**89.2%**	**89.2%**	89.2%	**89.2%**
Linear SVM L1 Penalty	88.04%	89.07%	87.28%	88.17%
SGD SVM L2 Peanlty	87.76%	87.28%	88.12%	87.7%
SGD SVM L1 Penalty	84.62%	82.48%	86.16%	84.28%
SGD SVM ElasticNet Penalty	86.43%	85.2%	87.33%	86.26%
Ridge Classifier	88.9%	88.65%	89.09%	88.87%
Preceptron Classifier	87.93%	87.87%	87.97%	87.92%
KNN	75.6%	66.6%	81.18%	73.2%
Random Forest	83.41%	84.45%	82.73%	83.58%
Bernoulii NB	78.96%	61.53%	**94.47%**	74.52%
Complment NB	87.34%	84.13%	89.9%	86.92%

Table (4) shows a comparison of our best result with other established methods including state of the art results. Despite not being competitive with state of the art results such as (Thongtan & Phienthrakul 2019), our model still shows good performance despite its simplicity compared to the models and methods listed in table (4). Our result outperforms even some complex models such as S-LSTM and CNN-LSTM systems that require more computational resource and time to train.

5. Conclusion

In this paper we explored different ML techniques to perform text classification for sentiment analysis. Despite its simplicity, traditional bag of n-gram models combined with SVM classifiers still provide adequate performance compared to other complex systems. Binary features provide consistent gain for classification when used with bag of n-grams. Higher order grams seem to provide considerable performance increase in sentiment classification.

6. Future Work

Exploring the use of PV and DV-ngram embeddings and testing their performance as features for classification. Using NB weighting of features and testing performance on regular ML classifiers.

Table 4. Results Comparison

Comparison with Results on IMDB Sentiment Analysis	
	accuracy
Our Approach	**89.2%**
Method	
S-LSTM (Zhang, Liu & Song 2018)	87.15%
CNN+LSTM (Camacho-Collados & Pilehvar 2018)	88.9%
W-Neural-BON Ensemble (Li et al., 2016)	93.51%
TGNR Ensemble (Li et al., 2017)	93.51%
TopicRNN (Dieng et al., 2017)	93.76%
One-hot bi-LSTM (Johnson & Zhang, 2016)	94.06%
Virtual Adversial (Miyato et al., 2016)	94.09%
BERT Large finetune UDA (Xie et al. 2019)	95.80%
NB-weighted-BON + DV-ngram (Thongtan & Phienthrakul 2019)	96.95%
NB-weighted-BON + L2R Dot Product (Thongtan & Phienthrakul 2019)	97.17%
NB-weighted-BON + Cosine Similarity (Thongtan & Phienthrakul 2019)	**97.42%**

References

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit.* "O'Reilly Media, Inc.".

Camacho-Collados, J. and Pilehvar, M. T. (2018). On the Role of Text Preprocessing in Neural Network Architectures: An Evaluation Study on Text Categorization and Sentiment Analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, (pp. 40–46), Brussels, Belgium. Association for Computational Linguistics. doi:10.18653/v1/W18-5406

Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805.*

Dieng, A. B., Wang, C., Gao, J. and Paisley, J. (2017). Topicrnn: A recurrent neural network with long-range semantic dependency. In *Proceedings of the 5th International Conference on Learning Representations.*

Hochreiter, S., & Schmidhuber. J. (1997). Long short-term memory. *Neural computation 9*, no. 8 (pp. 1735-1780). https://doi.org/10.1162/neco.1997.9.8.1735

Iyyer, M., Manjunatha, V., Boyd-Graber, J., and Daumé III, H. (2015). Deep unordered composition rivals syntactic methods for text classification*. In Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing* (volume 1: Long papers) (pp. 1681-1691). doi:10.3115/v1/P15-1162

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning* (pp. 137-142). Springer, Berlin, Heidelberg. https://doi.org/10.1007/BFb0026683

Johnson, R., and Zhang, T. (2016). Supervised and semi-supervised text categorization using lstm for region embeddings. In *Proceedings of the 4th Inter-national Conference on Learning Representations*.

Jurafsky, D., and Martin, J. H. (2022). *Speech and Language Processing* (3rd ed. draft).

Le, Q., and Mikolov, T. (2014). Distributed representations of sentences and documents*. In International conference on machine learning* (pp. 1188-1196). PMLR

Li, B., Liu, T., Du, X., Zhang, D., and Zhao, Z. (2015). Learning document embeddings by predicting n-grams for sentiment classification of long movie reviews. *arXiv preprint arXiv:1512.08183*.

Li, B., Zhao, Z., Liu, T., Wang, P., and Du, X. (2016). Weighted neural bag-of-n-grams model: New baselines for text classification. In *Proceedings of the 26th International Conference on Computational Linguistics*, (pp. 1591–1600).

Li, B., Liu, T., Zhao, Z., Wang, P. and Du, X. (2017). Neural bag-of-ngrams. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, (pp. 3067–3074)

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng A. Y. and Potts, C. (2011). Learning Word Vectors for Sentiment Analysis. In proceedings of *The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)* (pp. 142-150).

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), (pp. 39-41). https://doi.org/10.1145/219717.219748

Miyato, T., Dai A. M., and Goodfellow, I. (2016). Adversarial training methods for semi-supervised text classification. In *Proceedings of the 4th International Conference on Learning Representations*

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *The Journal of machine learning research*, *12*, (pp. 2825-2830).

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. *In Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10 (EMNLP '02)*. (pp. 79-86). Association for Computational Linguistics, USA. https://doi.org/10.3115/1118693.1118704

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543), Doha, Qatar. Association for Computational Linguistics. doi:10.3115/v1/D14-1162

Peters, E. M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018) Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. (pp. 2227–2237), New Orleans, Louisiana. Association for Computational Linguistics. doi:10.18653/v1/N18-1202

Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019). How to fine-tune bert for text classification?. In *China national conference on Chinese computational linguistics* (pp. 194-206). Springer, Cham. https://doi.org/10.1007/978-3-030-32381-3_16

Thongtan, T., and Phienthrakul, T. (2019). Sentiment classification using document embeddings trained with cosine similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop.* (pp. 407-414), Florence, Italy. Association for Computational Linguistics. doi:10.18653/v1/P19-2057

Turney, P. D. (2002). Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. *In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL '02)*. (pp. 417-424). Association for Computational Linguistics, USA. https://doi.org/10.3115/1073083.1073153

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Wang, S. I., and Manning, C. D. (2012). Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (pp. 90-94).

Xie, Q., Dai, Z., Hovy, E., Lu-ong, M. and Quoc V. Le. (2019). Unsupervised data augmentation. *arXiv preprint arXiv: 1904.12848*.

Zhang, Y., Liu, Q., and Song, L. (2018) Sentence-State LSTM for Text Representation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 317-327) Melbourne, Australia. Association for Computational Linguistics. doi:10.18653/v1/P18-1030