COMBINING WORD EMBEDDINGS FOR BINARY CLASSIFICATION TASKS

A PREPRINT

Andrew Schwartz
Denver, CO
andrewtschwartz@acm.org

January 31, 2020

ABSTRACT

This article explores the potential of combining two word-level vector representations, or word embeddings, for binary classification Natural Language Processing tasks. This research employs a narrow scope, considering GloVe, ELMo, and BERT embeddings and testing on a single data set. This article compares the classification ability of a single embedding with the classification ability of the same embedding and an additional embedding. Specifically, this article finds that combining adding ELMo to BERT (or BERT to ELMo) does not increase classification ability over ELMo alone or BERT alone. Additionally, this article finds that combining contextual (ELMo) or bidirectional (BERT) embeddings with static embeddings (GloVe) does not increase classification ability, and that combining ELMo embeddings with GloVe embeddings or BERT embeddings with GloVe embeddings increases classification performance over GloVe embeddings alone.

***K*eywords** Word embeddings · Natural Language Processing · Binary Classification

1 Introduction

The field of natural language processing (NLP) depends on the quality of word embeddings, such as Word2Vec [1], FastText [2], and BERT [3] to mathematically represent text. Modern NLP libraries offer the ability to combine (or "stack") word embeddings through vector concatenation [4], claiming to achieve state-of-the-art performance. While research has been performed on the potential for combined embeddings for select NLP tasks [5], there has yet to be an evaluation of the efficacy of combining embeddings for binary classification tasks. This paper offers an exploration of the potential for combining word vectors for binary classification tasks in NLP. Specifically, this article explores combinations of two embeddings and tests them on a single data set.

2 Methods

2.1 NLP Library

The training for each model was performed using the state-of-the-art Flair Natural Language Processing library's Recurrent Neural Networks (Flair NLP, [4]), with a 70%/10%/20% training/development/testing split of the data set. More information about the model training can be found in Section 2.2.1.

Flair was configured to stop training at 150 epochs or once the learning rate α reached less than 0.0001.

2.2 Embeddings

For this analysis, the following embeddings were used for comparison:

- GloVe [6]

- ELMo [7]
- BERT [3]

All of these embeddings operate at the word level, but only ELMo and BERT are contextual embeddings, and BERT is the only bidirectional embedding.

2.2.1 Model Training

Pretrained models from the Flair NLP library were used for this research, with the following details. All of the pretrained models are the defaults in Flair NLP, and are supported by the library.

- *GloVe* The pretrained GloVe model uses 6 billion tokens and 400,000 unique words, trained on uncased English text.
- *ELMo* The pretrained ELMo model uses 4096 hidden units, 2 layers, and 93.6 million parameters.
- *BERT* The pretrained BERT model uses 12 layers, 768 hidden units, 12-heads, and 110 million parameters, trained on lowercase English text.

2.3 Classification Task

For this analysis, the SMS Spam Collection Data Set [8] was used, a collection of $n = 5574$ text messages, where each message is classified as either "spam" or "ham" (not spam). The classification task is to classify any given text message as spam or not spam.

2.4 Determining Statistical Significance While Comparing Models

2.4.1 Choosing the corrected McNemar's Test for Determining Statistical Significance

The corrected McNemar's test [9, 10] was chosen instead of a t-test to compare models since the models are trained on the same data and thus violate the t-test assumption of independence and the classification task can only be performed once. [11] The significance test is performed to determine the statistical significance between a model X with one embedding $X = M(e_1)$ and a model Y with the same embedding, and an additional embedding $Y = M(e_1, e_2)$. This test has been determined to be suitable for classification algorithms that can only be run once and will result in a low Type I error [12].
For this analysis, the null hypothesis is that there is no statistically significant difference between the model with a single embedding and the model with one additional embedding. The alternative hypothesis is that there exists a statistically significant difference between these same two models. Thus,

- $H_0 : X = Y$ (There is not a statistically significant difference between X and Y.)
- $H_1 : X \neq Y$ (There exists a statistically significant difference between X and Y.)

The corrected McNemar's test was performed at a significance level $\alpha = 0.05$, as reflected by binary classification experiments that use McNemar's test in the literature. [11, 13]
(Note that the method for combining embeddings for $M(e_1, e_2)$ is to concatenate the b-dimensional vector e_2 to the a-dimensional vector e_1 to create a single $(a + b)$-dimensional vector representing both e_1 and e_2.) [4]

2.4.2 Corrected McNemar's Test and Contingency Tables

Consider, as an example, the classifiers A and B, where $A = M(e_1)$ (A is a model that is a function of embedding one) and $B = M(e_1, e_2)$ (B is a model that is a function of both embedding one and embedding two). Imagine that Table 1 represents the results of these two hypothetical classifiers.

Using the information in Table 1, we can generate a 2×2 contingency table for these two models. A generic form of this contingency table is shown in Table 2, and Table 3 shows a specific form of the contingency table, based on the data in Table 1.

$$\chi^2 = \frac{(\mid B - C \mid -1)^2}{(B + C)} \tag{1}$$

Data ID	Model A Result	Model B Result	Actual Class	A Correct	B Correct
1	pos	pos	pos	true	true
2	pos	neg	pos	**true**	**false**
3	neg	neg	neg	true	true
4	neg	neg	neg	true	true
5	pos	pos	neg	false	false
6	pos	neg	pos	**true**	**false**
7	neg	pos	pos	**false**	**true**
8	neg	neg	neg	true	true
9	pos	pos	pos	true	true
10	neg	pos	pos	**false**	**true**
11	pos	neg	pos	**true**	**false**

Table 1: Comparison of the classification of (hypothetical) Model A and Model B. Bolded values represent differently classified results across A and B.

	X Correct	**X Incorrect**
Y Correct	A	B
Y Incorrect	C	D

Table 2: Generic contingency table for Models A and B

Using the hypothetical contingency table for models A and B in Table 3 and the equation for the corrected McNemar's statistic in Equation 1, we find that $\chi^2 = 0.2$, yielding a p-value of 1.0, far greater than $\alpha = 0.05$. Thus, in this hypothetical scenario, we would not reject the null hypothesis and would conclude that the addition of embedding e_2 has no effect of the classification ability of model B.

3 Results and Conclusions

For this research, ${}^3P_2 = 6$ model comparison experiments were performed, one for each combination of two embeddings. The results of the experiments, including p and α values and whether or not the null hypothesis was accepted is in Table 4.

Although this work offers some insight into the potential for combining word embeddings for binary classification NLP tasks, it does not offer any comprehensive conclusions about the performance differences between a model based on a single embedding and a model based on that same embedding and an additional embedding. This work, however, does indicate that for certain embedding combinations, there may be discernible differences in performance and for other combinations, there may be no discernible difference in performance.

3.1 Combining BERT and ELMo does not improve classification ability over either embedding alone

Adding BERT to a model based on ELMo or adding ELMo to a model based on BERT does not seem to increase performance over a single embedding alone. This may have significant implications for researchers looking to achieve state-of-the-art results since it indicates that opting to include BERT along with ELMo (or vice versa) does not actually increase performance [1] yet does significantly increase computational complexity. (This is especially true for BERT because of its bidirectional embeddings.)

[1] See section 4.1.2 for information about the closeness of the p and α values for comparing $M(ELMo)$ and $M(ELMo, BERT)$. It is possible that, given a different dataset, the p for this test could be below α.

	A Correct	**A Incorrect**
B Correct	5	2
B Incorrect	3	1

Table 3: Contingency table for Models A and B, based on Table 1

Embedding Combination Info		McNemar's Results		
Base Model	Combined Model	χ^2	p	Result
$M(GloVe)$	$M(GloVe, ELMo)$	12.00	5.32E-04	H_1
$M(GloVe)$	$M(GloVe, BERT)$	16.69	4.40E-05	H_1
$M(ELMo)$	$M(ELMo, GloVe)$	1.50	2.21E-01	H_0
$M(ELMo)$	$M(ELMo, BERT)$	3.06	8.01E-02	H_0
$M(BERT)$	$M(BERT, ELMo)$	0.50	4.80E-01	H_0
$M(BERT)$	$M(BERT, GloVe)$	0.50	4.80E-01	H_0

Table 4: Results of combined embeddings experiments, with p values and χ^2 results.

3.2 Including GloVe with a more complex embedding does not increase classification ability

Combining GloVe with ELMo or GloVe with BERT does not seem to increase performance ability over ELMo or BERT alone. For this reason, this work indicates that the additional (albeit marginal) computational complexity created by opting for a model $M(GloVe, ELMo)$ or $M(GloVe, BERT)$ instead of $M(ELMo)$ or $M(BERT)$ is a waste as it does not increase classification ability. This is likely due to the complex nature of ELMo and BERT in comparison to GloVe: since ELMo and BERT are contextual embeddings (and are thus more complex than GloVe), including GloVe may not matter if a model is already using one a more complex embedding.

3.3 Adding BERT or ELMo to GloVe embeddings improves classification ability

Combining BERT with GloVe or ELMo with GloVe yeilds better results than GloVe on its own. This is hardly surprising since the technological improvements such as the consideration of context and bidirectionality are present in ELMo and BERT, but not in GloVe.

4 Limitations and Future Suggestions

4.1 Limitations

This research explores a very limited scope in evaluating combined embeddings: it exclusively explores a single NLP task (binary classification), compares only a small set of embeddings ($n = 3$), trains on a single data set, and only compares combinations of two simultaneous embeddings (as opposed to more than two embeddings).

4.1.1 SMS Dataset

This research considers combinations of embeddings tested on a single dataset of text messages. This particular corpus (the SMS Spam Collection Data Set) is "rife with idioms and abbreviations," [8] which means that this corpus doesn't parallel traditional English text. This could mean that these particular conclusions do not hold for English text.

4.1.2 Comparing ELMo vs. ELMo and BERT

Considering the result of the corrected McNemar's test comparing $M(ELMo)$ and $M(ELMo, BERT)$, with $p \approx 0.08$, it is possible that a p value below the $\alpha = 0.05$ threshold could be achieved on a different data set. If this were to be the case, the null hypothesis would be rejected and we would conclude that the addition of the BERT embedding to ELMo does improve the classification performance. Future work could explore the addition of BERT to a model using ELMo, to offer more insight into this particular combination of embeddings.

4.2 Suggestions for Future Work

Considering that this research explores a very limited scope of applications for combined embeddings, there are many options for relevant future work.

4.2.1 Considering combinations of more than two stacked embeddings

One option for future research would be to perform a more in-depth analysis of combined embeddings for binary classification models, evaluating combinations of more than two simultaneous embeddings (i.e. considering models of the form $M(e_1, e_2, ..., e_n)$ where $n > 2$) and considering a larger set of embeddings. This research might shed light on optimal embedding combinations for binary classification problems and could draw conclusions on combining

different types of embeddings, such as the effects of combining a contextual embedding with a non-contextual one, or a bidirectional embedding with a contextual one, etc.

4.2.2 Considering evaluating embedding combinations on multiple datasets

Another consideration for future research would be to execute this work's methods on multiple binary classification data sets. One limitation of this work is that it explores the efficacy of stacked embeddings on a single data set, but by training models with stacked embeddings on multiple data sets, it could be possible to reach different conclusions and offer a more complete analysis of stacked embeddings for binary classification tasks.

4.2.3 Considering combined embeddings for other NLP tasks

Lastly, future work could take on a much larger scope and consider the possibilities of combined embeddings for multiple NLP tasks. Although [5] explores combining embeddings for Part-Of-Speech Tagging, Chunking, Named Entity Recognition, and Mention Detection, it does not investigate the potential for combined embeddings for classification tasks or many other NLP tasks.

References

[1] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[2] Ben Athiwaratkun, Andrew Gordon Wilson, and Anima Anandkumar. Probabilistic fasttext for multi-sense word embeddings. *arXiv preprint arXiv:1806.02901*, 2018.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[4] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.

[5] Sahar Ghannay, Benoit Favre, Yannick Esteve, and Nathalie Camelin. Word embedding evaluation and combination. In *LREC*, pages 300–305, 2016.

[6] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[7] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proc. of NAACL*, 2018.

[8] Tiago A Almeida, José María G Hidalgo, and Akebo Yamakami. Contributions to the study of sms spam filtering: new collection and results. In *Proceedings of the 11th ACM symposium on Document engineering*, pages 259–262. ACM, 2011.

[9] Allen L Edwards. Note on the "correction for continuity" in testing the significance of the difference between correlated proportions. *Psychometrika*, 13(3):185–187, 1948.

[10] Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, Jun 1947.

[11] Thomas G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923, 1998.

[12] Johannes Fürnkranz. Round robin classification. *Journal of Machine Learning Research*, 2(Mar):721–747, 2002.

[13] Betul Erdogdu Sakar, M. Erdem Isenkul, C. Okan Sakar, Ahmet Sertbas, Fikret Gurgen, Sakir Delil, Hulya Apaydin, and Olcay Kursun. Collection and Analysis of a Parkinson Speech Dataset With Multiple Types of Sound Recordings. *IEEE Journal of Biomedical and Health Informatics*, 17(4):828–834, July 2013.