Comparative Performance of Popular Text Embeddings in Twitter Disaster Prediction

Thomas Hamnett
Master of Information & Data Science
University of California - Berkeley
thamnett@berkeley.edu

Adrian Lam
Master of Information & Data Science
University of California - Berkeley
ajlam@berkeley.edu

Abstract

In recent years, Natural Language Processing model performance achieved state-of-the-art performance via the development and adoption of pre-trained embedding models that encode words and their contexts within sentences into dense, representative vectors rather than sparse, bag-of-word token counts. While there is much existing literature on which neural network architectures best optimize performance for different tasks (Zhang and Wallace, 2016), and on the performance improvements achieved by each new pre-trained embedding model (Peters et al, 2018), there exists little research on comparative performance among between different state-of-the-art pre-trained word embeddings. Additionally, there are only a small number of papers addressing the impact of pre-processing on performance of these models. We thus conduct an extensive analysis of comparative performance in a sentence classification task, between models that combine popular pre-trained embeddings with four different levels of text pre-processing. We conclude that Google's Universal Sentence Encoder (USE) and Embeddings from Language Models (ELMo) perform the best among our candidate models, and that performance was maximized with only basic pre-processing.

1 Introduction

Sentence classification is a popular use case for Natural Language Processing, and is used widely in such applications as Sentiment Analysis and Part-Of-Speech tagging models. Our research explores sentence classification using data from another popular area with NLP research: Twitter. Specifically, our research aims to predict which Tweets with disaster-related hashtags represent true disasters vs. false alarms. This research has public safety implications for first responder agencies to detect disasters in a more accurate and timely manner, using Twitter data as an early warning system.

Modeling natural language requires representations of corpora numerically. Early simple methods of embedding text (e.g. one-hot encoding) have seen substantial advances, leading to improved representations of language and performance in such tasks as classification, translation, and machine communication. Some of the most successful representations, built from massive corpora and leveraging deep learning, have been made available as pre-trained word and sentence embeddings.

In this paper, we explore relative performance of several state-of-the-art

pre-trained embeddings in sentence prediction on our Twitter dataset. We also compare the impact of different levels of preprocessing prior to learning embeddings. And finally we explore ways to optimize performance by creating ensemble models among the highest performing individual embeddings. The results of our research help inform which embeddings are most performant on Twitter data, how preprocessing impacts performance of pre-trained embeddings, and the scale of performance improvements possible via ensemble models using multiple embeddings.

The hypotheses driving our approach are:

- Sentence and contextual word embeddings models will outperform word embedding models due to better capturing context of how individual words are being used
- BERT will perform best among our models due to its approach of combining bi-LSTM architecture (as in ELMo) with transformer encoders (as in USE)
- Some light pre-processing will improve model performance due to anomalies inherent to Twitter text
- Performance improvements can be achieved via an ensemble model of multiple embeddings

2 Deep learning NLP models

Deep learning has been fundamental to recent advances in NLP but practitioners face several challenges in building and deploying successful language models. Two major challenges are 1) collecting sufficiently large corpora to build a robust language model, and 2) the large amount of computing power necessary to train models on such corpora. Pre-trained models address these challenges as they are typically developed by well-funded academic and/or private organizations with the resources to collect large amounts of data and power computation of models. Additionally, the organizations attract top NLP talent who apply leading-edge techniques in their modeling.

We compare performance of several of these pre-trained models, each with distinct architectures (OneHot and GloVe word embeddings; ELMo, USE, BERT sentence embeddings). Our initial hypothesis was that sentence and dynamic embeddings would perform better than static embeddings due to capturing additional important context and that BERT would perform the best due to its combining the key features of both ELMo (bi-LSTM) and USE (transformer-encoder).

Our results are (discussed later in depth) largely in-line with our hypotheses, with a few areas of deviation. Sentence and dynamic embeddings outperform static word embeddings, and BERT performs relatively well. However, we find that the USE (Transformer) and ELMo models slightly outperform BERT. Since USE and ELMo each performed equally well, we do not have definitive evidence that either transformer-encoder models such as USE or bi-LSTM architectures such as that of ELMo are better suited as embeddings for Twitter data.

3 Dataset

We obtained from Kaggle a dataset composed of 7,613 Tweets with hashtags related to disasters. The Tweets were labeled to indicate whether the Tweet was associated with a real disaster or not, and included two additional fields to specify the keywords and the location associated with the Tweet.

Successful classification of Tweets as real disasters or not has important public safety implications. First responder agencies may be able to use Twitter as an early warning system to identify emerging disasters and more quickly deploy resources - while also identifying false alarm situations that can better manage first responder resources. Such a model could ultimately minimize damage and suffering resulting from a potential disaster and reduce costs associated with responding to false alarms.

Twitter data has unique challenges for NLP as it often contains non-standard language. Examples include typos, @s, URL links, abbreviations, acronyms, colloquialisms, and incorrect grammar. Because of these challenges, we look at various levels of preprocessing to determine its impact on model performance.

4 Text Preprocessing

Data preprocessing is a common step in any machine learning models, and with our various pre-trained word embeddings, we wanted to explore how each embedding reacted to different methods of preprocessing as well. Our initial hypothesis was some light preprocessing would improve our performance to correct for the previously discussed challenges inherent in Twitter data. However, as detailed in the results section, we find that no pre-processing performed nearly the same or better than other pre-processing techniques. We applied 4 different types of preprocessing to our training data, to examine how much assistance, or lack thereof, preprocessing provided to our models.

Each embedding was trained on the following 4 types of preprocessing:
1) Original - inputs the Tweets as-is
2) Basic - applies a standard lowercasing and punctuation removal
3) PortStem - applies a more advanced set of rules to remove common stopwords, word endings, and uses the base form of words

```
explore_Preprop(65)
```

```
Original -  @nxwestmidlands huge fire at Wholesale markets ablaze http://t.co/rwzbFVNXER
Basic -  nxwestmidlands huge fire at wholesale markets ablaze httptcorwzbfvnxer
PortStem -  nxwestmidland huge fire wholesal market ablaz httptcorwzbfvnxer
Smart -   huge fire at wholesale markets ablaze
```

Figure 1. Various preprocessing outputs of the same input text

4) Smart - a final preprocessing approach our team generated as a result of EDA. From our initial EDA, we noticed that the nature of Twitter data includes external links and @ tags of other people, which may be useful to users with the ability to click and further delve into hyperlinks, but for our use case of sentence classification, these were potentially extraneous and may serve to distract rather than inform.

Kutuzov et al. 2019 finds that lemmatizing English language text does not improve ELMo performance, contributing to our hypothesis that only 'light' pre-processing would be beneficial due to Twitter text anomalies.

In Figure 1, we see the effect of our 4 types of preprocessing on one of the training set input text data.

5 Embeddings

Recent innovations in word and sentence embeddings have been enabled by advances in using neural networks for natural language processing tasks. In its early variants, NLP models used basic count of words (or CBOW) in order to represent the contents of a sentence, and while they proved to be effective, a lot of contextual information about word use was lost and CBOW required massive, sparse matrices, with dimensions equalling the length of the corpus vocabulary.

More recently, researchers have developed pre-trained word embedding models on massive amounts of text to help transfer learned relationships between words into encoding new sentences. These pre-trained word representations (Mikolov et al, 2014) help model complex characteristics as well as relationships between words, and help tokenize text that otherwise couldn't be read in as inputs to Neural Networks. We use the following embeddings to compare performance:

- OneHot, Self-trained Embeddings: Simple counts on padded sentences
- GloVe: Word vector matrix trained through an unsupervised learning algorithm from Stanford
- ELMo: Deep contextualized word representations (Peters et al, 2018) with dynamic embeddings based on usage and context, using bi-LSTM
- BERT: State-of-the-art model combining bi-LSTM with transformer-encoder architecture via use of a masked language model
- USE: Sentence encoder with DAN (averaged output) and Transformer (more computationally intensive) variants, with transformer-encoder architecture

6 Results

6.1 Baseline model

We consider two baseline models:

1) Basic baseline model
Most common class prediction.
Baseline accuracy = 57.0%

2) Logistic regression baseline model
Logistic regression on ELMo embeddings with no pre-processing, to set a realistic threshold on which to improve.
Baseline accuracy = 79.1%
Baseline F-1 score = 0.741

6.2 Results: Combinations of Embeddings and Preprocessing

	Original	Basic	PortStem	Smart
OneHot	**66.12%**	65.73%	65.40%	64.54%
GloVe	80.37%	79.51%	78.27%	**80.43%**
ELMo	81.81%	**81.88%**	79.91%	81.75%
USE (DAN)	**81.94%**	**81.94%**	79.05%	81.88%
USE (Trans-former)	**82.27%**	82.21%	79.71%	81.75%
BERT	82.21%	**82.86%**	80.70%	82.27%

Figure 2. Output accuracies of different embeddings

	Original	Basic	PortStem	Smart
OneHot	**0.490**	0.410	0.434	0.385
GloVe	**0.757**	0.747	0.733	0.755
ELMo	0.775	**0.784**	0.758	0.777
USE (DAN)	**0.783**	**0.783**	0.746	0.782
USE (Trans-former)	0.783	**0.784**	0.753	0.777
BERT	0.779	**0.781**	0.761	0.778

Figure 3. Output accuracies of different embeddings

The results of our embedding and preprocessing combinations show:

- All of our models except OneHot (and GloVe with PortStem) beat Baseline logistic regression model F-1 score, suggesting neural nets with pre-trained embeddings are a valid approach for this task
- USE and ELMo had the highest F-1 scores (despite BERT having the highest accuracy)
- Model performance (F-1 score) was highest with Basic pre-processing for sentence embeddings and Original for word embeddings, suggesting whole Tweets benefit from light pre-processing
- PortStem pre-processing had the worst performance for all embeddings except OneHot, suggesting such 'heavy' pre-processing eliminates important signal from pre-trained embeddings

6.3 Ensemble Models

In addition to the individual models we compared, we explored the question if performance could be improved from an ensemble of multiple embeddings. Our hypothesis was that an ensemble could outperform the individual embeddings, drawing on the strengths of different architectures. To test this hypothesis, we created an ensemble model that combines outputs from three neural net models into a final Dense layer for prediction. We use a combination of USE, BERT, and GloVe as our test case, combining both word and sentence embedding inputs. And we use Basic pre-processing for each. Our results confirm our hypothesis, with the ensemble model outperforming any of the individual embedding models:

Accuracy = 84.11%
F-1 Score: 0.799

6.4 Model Architecture Considerations

Our models run on 'small' data with only 7,613 records in the dataset. We find that achieving reasonable performance has the following implications on hyperparameters:

- Models must be small: Number of layers needs to be minimal (e.g. two layers) and number of internal nodes must likewise be small (e.g. dense layers with 8 internal nodes)
- Learning rate must be low (e.g. 0.0001 in Adam optimizer)
- Regularization/Dropout may improve performance (used in the more complex ensemble model)

These restrictions on hyperparameters were necessary to avoid severe overfitting, which we witnessed when making even small adjustments that increased model complexity. We believe that a larger dataset could have afforded more complex models and is a possible extension of this research.

7 Conclusion

In summary, our research concludes the following:

- ELMo and USE both showed the best performance in Twitter disaster sentence classification compared to other candidate embedding models, highlighting the strength of each pre-trained embedding model despite their different architectures
- Little or no pre-processing performed much better than 'heavy' pre-processing (e.g. stemming, lemmatization) in models using pre-trained embeddings, reinforcing the idea that pre-processing text input when using pre-trained embeddings eliminates valuable signal
- While baseline logistic regression model performs relatively well, our neural network models produce non-negligible performance improvements, reinforcing their usefulness in NLP tasks
- Combining embeddings that perform well individually into an ensemble model can generate performance superior to any of the individual embedding models

References

[Cer et al.2018] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Ha et al. 2018. Universal Sentence Encoder. https://arxiv.org/pdf/1803.11175.pdf

[Hassan et al.2019] Hebatallah A Mohamed Hassan, Giuseppe Sansonetti, Fabio Gasparetti et al. 2019. BERT, ELMo, USE and InferSent Sentence Encoders: The Panacea for Research-Paper Recommendation? http://ceur-ws.org/Vol-2431/paper2.pdf

[Kim. 2014] Yoon Kim. 2018. Convolutional Neural Networks for Sentence Classification. https://www.aclweb.org/anthology/D14-1181.pdf

[Kutuzov et al. 2019] Andrey Kutuzov and Elizaveta Kuzmenko. 2019. To lemmatize or not to lemmatize: how word normalisation affects ELMo performance in word sense disambiguation. https://arxiv.org/pdf/1909.03135.pdf

[Mikolov et al.2013] Tomas Mikolov, Ilya Sutskever, Kai Chen et al. 2013. Distributed Representations of Words and Phrases and their Compositionality. https://arxiv.org/pdf/1310.4546.pdf

[Peters et al.2018] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner et al. 2018. Deep Contextualized Word Representations. https://arxiv.org/pdf/1802.05365.pdf

[Zhang et al.2016] Ye Zhang and Byron C. Wallace. 2016. A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification. https://arxiv.org/pdf/1510.03820.pdf