

Disaster Tweet Classification: A Comparative Analysis of Conventional and Deep Neural Network Machine Learning Models

Alexis Wu

Princeton University
alexiswu@princeton.edu

Abstract

During disasters, Twitter/X can bridge affected civilians and relief agencies through real-time dissemination of information. The scale at which Twitter users produce data necessitates the development of machine learning models that accurately classify disaster-related tweets for efficient response coordination and resource allocation. In this work, we aim to assess the effectiveness of various Natural Language Processing techniques in distinguishing disaster-related tweets from others. We investigate the performance of 6 conventional classifiers and 2 deep neural network models, specifically bidirectional LSTMs, on a popular Kaggle dataset comprised of English-language disaster-related tweets. We find that most of the conventional models suffer from having at least one considerably lower result, with recall falling in the range 0.63 – 0.71 for disaster-related tweets. Moreover, we find that our LSTMs achieve higher and more tightly clustered precision, recall, F1, and accuracy, indicating overall better performance. We contribute to existing research studying the application of classifiers to domain-specific datasets to aid in the selection of classifiers best suited for the task at hand.

1 Introduction

Social media, particularly Twitter, has become a platform for people to make announcements and discuss unfolding events live. Given how pervasive social media is today, organizations and agencies have paid increasingly more attention to Twitter users regarding keywords and hot topics.

Such a category of tweets includes “disaster tweets,” which consists of user tweets that discuss or raise awareness about real-world emergencies. Examples of disasters include natural disasters, such as earthquakes, wildfires, and hurricanes, as well as large-scale accidents. Agencies that pay particular attention to these tweets include disaster relief organizations like the Australian Red Cross

(Twitter, 2022), which are interested in applying automatic or machine learning techniques to identifying and classifying tweets as disaster-related. A key issue is that the use of metaphor and sarcasm by Twitter users makes it more challenging for text classifiers to determine whether the tweet is truly about a disaster or not.

This project’s research goal is to compare the efficacy of different Natural Language Processing methods in predicting whether or not a tweet discusses a disaster. To do so, we examine both conventional and neural network-based approaches. We follow the framework established by Kumar et al. (2019) to select 6 conventional models (Logistic Regression, Naive Bayes, Support Vector Machine (SVM), Decision Trees, Random Forest, and Gradient Boosting) and 2 different architectures for Long Short-Term Memory (LSTM). Given time constraints, we focus only on LSTMs, which are designed to take into account temporal properties of language (Hochreiter and Schmidhuber, 1997) and have been shown to perform better than vanilla feed-forward neural networks and traditional models with Bag of Words representations, especially on shorter text (Nowak et al., 2017).

2 Related Work

A wealth of literature has been interested in exploring the efficacy of both traditional machine learning models and emerging deep neural network approaches with respect to classification tasks. More recently, with the prominence of social media apps like Twitter, research has looked to apply these methodologies to tweet classification tasks. However, research often focuses on the performance evaluation of machine learning approaches for binary tasks like sentiment analysis, and less literature exists on classifying tweets as informative or uninformative.

One introductory paper that identifies this gap

in the literature conducts performance evaluation and compares Naive Bayes and Support Vector Machine (SVM), using mined text from tweets during the 2012 flooding in Metro Manila, a region in the Philippines (Parilla-Ferrer et al., 2014). This paper uses 612,622 tweets collected by the researchers of Ateneo de Manila University, from which a sample of 4,000 tweets were randomly selected and manually classified. Preprocessing tasks involved tokenization, stop word removal, and stemming with Porter’s algorithm. Moreover, the paper adopted the Bag of Words (BoW) technique to produce the main features in the word vector, where words with less than two characters were pruned. The paper follows the following supervised learning approach: to generate a model, the authors ran first Naive Bayes and then SVM learning algorithms on the training dataset to map the feature values (\mathbf{x}) to classes (y). Then, the learned model was applied to the test dataset to evaluate performance. Evaluation metrics included accuracy, recall, precision, AUC, and F-measure. The paper found that SVM had a statistically significant higher accuracy, AUC, recall, and F-measure, while Naive Bayes had higher precision.

A limitation of this paper was that it relied on manual classification of tweets by 3 separate annotators for English ($n = 1,563$ tweets), Tagalog ($n = 1,393$ tweets), and combination tweets ($n = 913$ tweets), which may have restricted the number of labeled tweets it could have produced and used. Overall, the paper confirmed findings from previous studies that found SVM to significantly outperform Naive Bayes when classifying short, unstructured, noisy text such as tweets.

A 2019 paper conducted a more holistic comparative analysis of machine learning models for disaster-related tweet classification (Kumar et al., 2019). The paper first examines 7 conventional machine learning models (Logistic Regression, Naive Bayes, SVM, K-Nearest Neighbors (KNN), Decision Trees, Random Forest, and Gradient Boosting) with 1-gram, 2-gram, and 3-gram Term Frequency-Inverse Document Frequency (TF-IDF). Then, it examines the performance of 5 deep learning models (Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Bi-directional GRU, and GRU-CNN with 100-dimensional GloVe word embeddings and 300-dimensional Crisis embeddings. For each of the neural network models, the loss function was categorical cross-entropy, the optimizer was Adam,

each hidden layer used ReLU activation, and the output layer used softmax activation.

Unlike the previous paper, which focused on the Habagat monsoon, this paper trains and tests models on four separate disaster event datasets split into Hurricane ($n = 4,087$ tweets), Earthquake ($n = 747$ tweets), Flood ($n = 197$ tweets), and Wildfire ($n = 528$ tweets). Data cleaning involved deleting duplicate tweet text, removing punctuation (i.e. @, &, #, !, %), and all text was made lowercase. The paper used a 75/25% dataset split into training and testing data. Evaluation metrics included precision, recall, and F1-score.

The paper found that among the seven conventional ML models, Gradient Boosting outperformed all other models in each disaster type, whereas SVM had the worst performance. Moreover, the results found that Crisis embeddings allowed for better performance for the Earthquake dataset, GloVe embeddings performed better on Wildfire, GloVe and Crisis were comparable on Flood, and there was mixed performance on Hurricane. Finally, the paper found that the five deep neural network-based models outperformed the seven conventional models. Amongst the five deep NN-based models, Bi-GRU performed best on Hurricane, GRU-CNN performed best on Earthquake, LSTM and GRU equally performed best on Flood, and GRU-CNN performed best on Wildfire.

Some strengths include the number of different models that were examined, which allowed for a broader understanding of different models when applied to the same task. Some shortcomings were that for the deep NN-based models, the authors used fixed batch size, learning rate, and optimizer, which are all hyperparameters that could be tuned for better performance. Moreover, the datasets on which the models were trained still seem to be quite small (with under 200 tweets). Overall, the paper identified certain combinations of models and word representations that performed better on different tasks, suggesting the need for task-specific design of machine learning models.

We situate our work within the context of these papers, applying both conventional and deep neural network-based machine learning approaches to a new dataset of disaster tweets, which we discuss more in the following Data section.

3 Data

To address our research problem, we needed a selection of tweets, labeled with whether or not they constitute a disaster. We intend for disaster-related tweets to encompass the following definition: tweets that discuss or capture a sudden event that causes great damage or loss of life, particularly due to natural catastrophes or accidents. Furthermore, we would like for our dataset to include tweets with figurative language or sarcasm, which may be easy for a human reader to spot, but challenging for a machine learning model to correctly classify. Please see Appendix A.1 for an example.

Dataset

We use a dataset created by Figure Eight (a company that previously shared this dataset on their Data For Everyone website), which was then made publicly available as a competition dataset on Kaggle in 2019 (Howard et al., 2019). The dataset contains a total of 10,876 tweets, with a training set ($n = 7,613$) that was hand-classified tweets labeled either 0 (not disaster-related) or 1 (disaster-related) and an unlabeled testing set ($n = 3,263$). Since the provided testing dataset is unlabeled, we rename the provided training set as “the dataset” and disregard the provided testing set.

Each data sample in the dataset includes a unique identifier (integer), the text of the tweet, the label, and two optional fields, keyword and the location from which the tweet was sent. We only examine the text, and not the location or keyword attributes, since we found these fields to frequently be NaN. The text and target labels are sufficient, as the goal of the project is to predict whether or not the tweet is disaster-related without the context clues like images that aid a human reader.

Given the time needed for one researcher to label over 10k tweets, we chose to use an existing dataset with a good amount of visibility on Kaggle, which has almost 130,000 entrants at the time of writing this paper. Please see our Limitations section for our assessment of and concerns regarding the dataset.

Exploratory Data Analysis

We conduct a few initial analyses to better understand the dataset. Of the 7,613 tweets, 3,271 were labeled as disaster-related and 4,342 were labeled as not disaster-related. We want to see if the tweet lengths are comparable between the classes. Bro-

ken down by tweet type (disaster-related vs. not disaster-related), we observe that the average number of characters per tweet is slightly shorter for not disaster-related tweets, which is also confirmed by the following histogram. However, the dataset does not seem to exhibit an extraordinary amount of variation between classes.

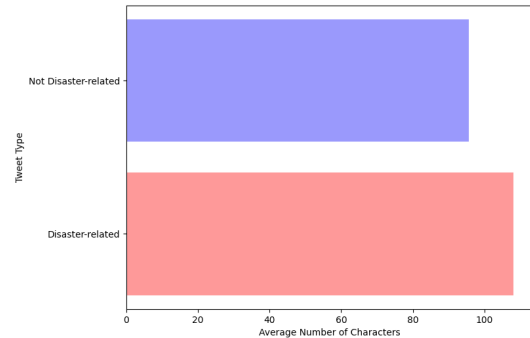


Figure 1: Average number of characters by tweet type

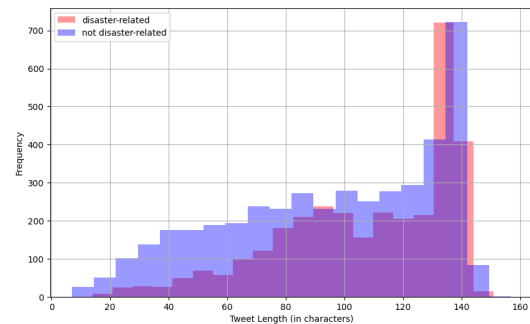


Figure 2: Distribution of Tweet Lengths by Tweet Type

4 Methodology

Text Preprocessing

We use a number of text preprocessing techniques to prepare and represent our dataset. First, we convert all text to lowercase. Then, we choose to remove words and elements in the dataset that do not carry much meaning and can interfere with what meaning our models learn from words. We remove stopwords, punctuation, user tags (e.g. @alexisjwu), HTML tags, links, and non-ASCII characters with Python’s Natural Language Toolkit (nltk) English corpus of stopwords and Snowball stemmer (GeeksforGeeks, 2022), both common techniques used in text preprocessing. The difference in performance between including and omitting numerical values was negligible, so we remove numerical values to reduce the dimensionality of the feature space and

simplify the data representation. Finally, we tokenize the text so each tweet becomes a list of tokens.

	text	target
0	[deeds, reason, earthquake, may, allah, forgiv...	1
1	[forest, fire, near, la, ronge, sask, canada]	1
2	[residents, asked, shelter, place, notified, o...	1
3	[people, receive, wildfires, evacuation, order...	1
4	[got, sent, photo, ruby, alaska, smoke, wildfi...	1

Figure 3: First five data samples, preprocessed

Conventional Models

For our conventional models, we take the preprocessed text and create a TF-IDF representation. We prefer TF-IDF to Bag of Words representations for the conventional models since we want to account for term importance in the corpus and perform normalization.¹ Furthermore, since Kumar et al. (2019) use TF-IDF representations, using TF-IDF would allow for a more comparable comparison between our works.

We select the following six conventional models to conduct our comparative analysis: Logistic Regression, Multinomial Naive Bayes, Support Vector Machine (SVM), Decision Trees, Random Forest, and Gradient Boosting. For SVM, we first try an SVM model with a linear kernel, since it is computationally efficient and often a good starting point. Then, we try using a Radial Basis Function (RBF) kernel. We typically use this kernel when the decision boundary is highly non-linear or not well-defined.

We choose to use a 60/20/20% training, validation, and testing dataset split. This means that we have 4,569 training data samples, and 1,522 validation and testing data samples each. We choose to use a validation set so that we can tune our hyperparameters before testing our model performance on the testing dataset.

Deep Neural Network Models

We aim to reproduce a subsection of Kumar et al. (2019)’s neural network methodology. Thus,

¹During implementation, we created both Bag of Words and TF-IDF representations and ran all six conventional models on each representation; we found that every model generally achieved higher performance on the TF-IDF representations. Please see Appendix A.2 for a table with the six conventional models’ performances with a Bag of Words representation.

we use GloVe word representations. We choose GloVe over Word2Vec, since Word2Vec and similar shallow window-based methods are limited in that scanning context windows fail to take advantage of repetition in data (Pennington et al., 2014). Moreover, GloVe outperforms Continuous Bag of Words (CBOW) Word2Vec while using a corpus less than half the size (Pennington et al., 2014).

As with the conventional models, for our LSTMs, we use a 60/20/20% training, validation, and testing dataset split.

We implement and analyze two bidirectional LSTMs: one baseline and one improved model. Prior to feeding our preprocessed text to our models, we download the GloVe Twitter word embeddings (glove-twitter-200, with 200-dimensional vector representations) to create the embedding matrix. Then, we convert the text data to sequences of integers as numerical representation for LSTM and pad sequences to a maximum fixed length to ensure that inputs have consistent shapes across batches when we use the Keras libraries.

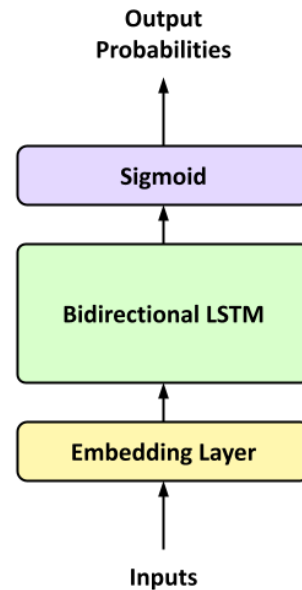


Figure 4: Layers in the baseline LSTM

To create our baseline bidirectional LSTM, we design a model with 1 embedding layer, 1 bidirectional LSTM layer (where the number of units equals the max sequence length), and 1 dense layer with sigmoid non-linear activation function (to reduce the number of units to 1 to output a single scalar value). We use binary cross-entropy loss, which is well-suited for a binary classification task

and is able to effectively handle class imbalance by penalizing the model more for misclassifying the minority class. We choose to use the Adam optimizer with a default learning rate of 0.01, as it converges faster than traditional stochastic gradient descent (SGD) methods and is the optimizer used in Kumar et al. (2019). We train the model for 8 epochs with a batch size of 32. While the model is fitting, we plot the training and validation accuracy, while also keeping track of training and validation loss, fine-tuning the model based on validation accuracy and loss.

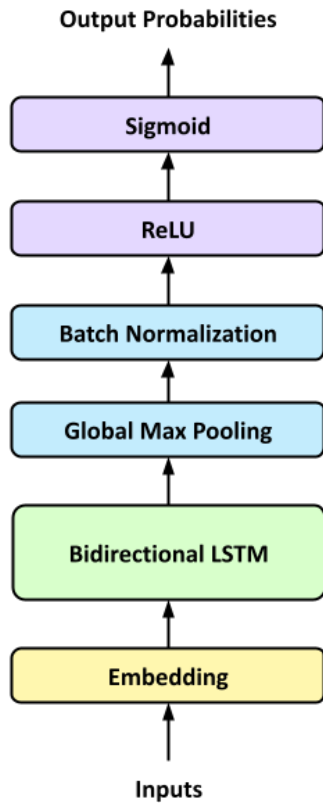


Figure 5: Layers in the improved LSTM

The architecture for our improved bidirectional LSTM was engineered after examining the performance of the baseline LSTM, which we discuss in the Results section. To create our improved LSTM, we design a model with 1 embedding layer, 1 bidirectional LSTM layer (where the number of units equals 128), 1 global max pooling layer, batch normalization, dropout, 1 dense layer with ReLU non-linear activation function (reducing the number of units to 32), dropout, and 1 dense layer with a sigmoid activation function. We still use binary cross-entropy loss, and experiment with other optimizers such as RMSProp. We train the model for

8 epochs with a batch size of 32. We mirror the approach for our baseline model by training the model for 8 epochs with a batch size of 32.

5 Results

For both our conventional models and our LSTMs, we measure performance in terms of precision, recall, and F1 score for both classes. We also include the accuracy for each model’s performance on the testing set.

5.1 Conventional Models

In Table 1, we show the results of our six conventional machine learning classifiers with TF-IDF. Examining the testing performance, we see that each model achieves higher recall for not disaster-related tweets (class = 0) than disaster-related tweets (class = 1). Moreover, we see that a majority of the models (with the exception of the Decision Trees classifier) generally achieves higher precision for disaster-related tweets than not disaster-related tweets.

More specifically, we find that Logistic Regression achieves the highest precision for disaster-related tweets (0.87), with Naive Bayes (0.86) and Random Forest (0.86) also achieving similar performance. Similarly, Logistic Regression achieves the highest recall for not disaster-related tweets (0.93), with Random Forest (0.92) and Naive Bayes (0.91) achieving similar performance. Naive Bayes achieved the highest F1 scores for both classes (0.85 and 0.76, respectively), suggesting that this model may best balance the precision-recall tradeoff.

Decision Trees had one of the worst performances on unseen data, with a precision of 0.70 and F1 score of 0.70 for disaster-related tweets and lower precision (0.77), recall (0.78), and F1 score (0.77) for not disaster-related tweets compared to the other five models. However, 4 of the 6 models—Logistic Regression (0.65), Naive Bayes (0.68), Random Forest (0.63), and Gradient Boosting (0.68)—all have lower recall scores for disaster-related tweets.

Regularization & Dimensionality Reduction

For our Logistic Regression model, we implement L2 (“Ridge”) Regularization to address model overfitting. After looping through a list of regularization strengths (denoted by C) from 0.01, 0.1, 0.25, 0.5, 1, 5, we find that model performance achieves the highest validation accuracy

TABLE 1
Results for Conventional Classifiers on Disaster Tweet Dataset (TF-IDF)

		Training			Validation			Testing			
Model	Class	<i>P</i>	<i>R</i>	<i>F1</i>	<i>P</i>	<i>R</i>	<i>F1</i>	<i>P</i>	<i>R</i>	<i>F1</i>	<i>Acc.</i>
Logistic Regression	0	0.86	0.97	0.91	0.78	0.93	0.85	0.78	0.93	0.85	0.81
	1	0.96	0.79	0.87	0.87	0.63	0.73	0.87	0.65	0.74	
Multinomial Naive Bayes	0	0.87	0.97	0.92	0.79	0.91	0.85	0.79	0.91	0.85	0.81
	1	0.96	0.81	0.88	0.85	0.65	0.74	0.86	0.68	0.76	
SVM (Linear Kernel)	0	0.92	0.98	0.95	0.78	0.89	0.83	0.80	0.89	0.84	0.81
	1	0.97	0.88	0.92	0.82	0.65	0.72	0.83	0.71	0.76	
Decision Trees	0	0.98	1.00	0.99	0.77	0.78	0.78	0.77	0.78	0.77	0.74
	1	1.00	0.97	0.99	0.69	0.67	0.68	0.70	0.70	0.70	
Random Forest	0	0.98	0.99	0.99	0.76	0.91	0.83	0.77	0.92	0.84	0.80
	1	0.99	0.98	0.99	0.84	0.61	0.70	0.86	0.63	0.73	
Gradient Boosting	0	0.89	1.00	0.94	0.77	0.88	0.82	0.78	0.87	0.82	0.78
	1	1.00	0.84	0.91	0.79	0.64	0.71	0.79	0.68	0.73	

TABLE 2
Results for LSTMs on Disaster Tweet Dataset (GloVe Embeddings)

		Testing			
Model	Class	<i>P</i>	<i>R</i>	<i>F1</i>	<i>Accuracy</i>
Baseline Bidirectional LSTM	0	0.79	0.84	0.81	0.78
	1	0.76	0.70	0.73	
Improved Bidirectional LSTM	0	0.81	0.90	0.85	0.82
	1	0.84	0.71	0.77	

TABLE 3
Results for Logistic Regression with Dimensionality Reduction Techniques

Model	Class	Testing			
		P	R	$F1$	Accuracy
Logistic Regression with SVD	0	0.79	0.87	0.83	0.79
	1	0.80	0.68	0.74	
Logistic Regression with PCA	0	0.80	0.91	0.85	0.80
	1	0.82	0.65	0.73	

of 0.8056 when $C = 1$. Validation performance decreases on either other side of $C = 1$; that is, neither decreasing nor increasing C results in a better model performance.

We then implement two dimensionality reduction techniques, Singular Value Decomposition (SVD) and Principal Component Analysis (PCA) to transform our original features into new sets of features that capture the most important information from our dataset. As seen in Table 3, these techniques do not actually provide considerable improvements from the vanilla Logistic Regression model. We further discuss this performance in the Discussion section.

5.2 Deep Neural Network Models

In Table 2, we show the testing performance of the two bidirectional LSTMs with GloVe word embeddings. Examining the testing performance, we see that each model still achieves higher recall for not disaster-related tweets (class = 0) than disaster-related tweets (class = 1). We discuss the models' performance on the training, validation, and testing data below.

With our baseline bidirectional LSTM, we see that the model overfits the training data, achieving a high training accuracy (0.97) and low training loss (0.06), but high validation loss and lower validation and testing accuracy. In fact, validation loss increased from 0.44 to 0.74 over the 8 epochs, while validation accuracy plateaus at a value around 0.78 to 0.79.

After examining our baseline LSTM, we identified the need to push the model away from overfitting the training data. Therefore, we implement a model with regularization (through dropout) and examine how it performs. We see that the model still performs best on the training data, achieving a high training accuracy (0.93) and relatively low training loss (0.17). For this model, validation loss is still quite high at around 0.45–0.55, but this is an improvement from the baseline model. Moreover, we achieve slightly higher validation and testing

accuracy, which are 0.81 and 0.82 respectively.

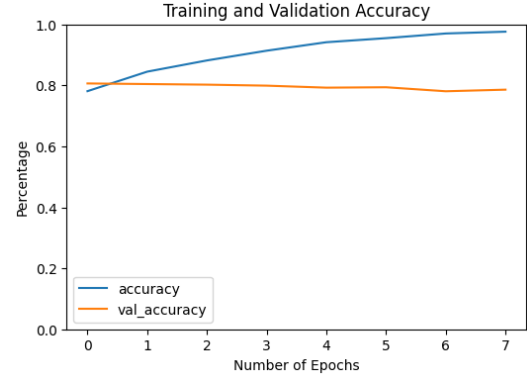


Figure 6: Baseline training and validation accuracy

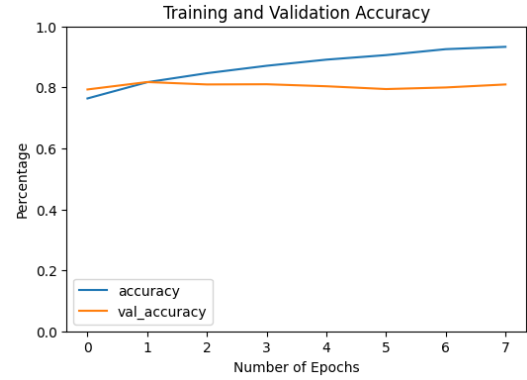


Figure 7: Baseline training and validation accuracy

Overall, we see the highest accuracy for our improved bidirectional LSTM out of all eight models. LSTMs also tend to have slightly less variability in terms of precision, recall, and F1 score between classes. For example, while Logistic Regression achieves a higher recall for not disaster-related tweets (0.93) than our improved bidirectional LSTM (0.90), our Logistic Regression model has a lower recall for disaster-related tweets (0.65) compared to the improved LSTM (0.71).

6 Discussion

We refrain from comparing our findings on model performance to the two papers examined in our literature review given our concerns outlined in the Discussion and Limitations sections that follow.

For our conventional models with TF-IDF, one possible explanation for the discrepancy between the high precision for disaster-related tweets and high recall for not disaster-related tweets is that these models may be more confident in predicting instances of class 0 as “not disaster-related” due to the higher frequency of these instances in the

dataset. When it comes to class 1, given fewer instances in the dataset, the model may be more cautious and less confident in its predictions, leading to higher precision but lower recall.

The Decision Trees classifier’s low performance likely occurs due to high overfitting on the training dataset, resulting in a model that does not truly learn the relationships between words in the dataset. This overfitting on the training dataset, with weak performance on the validation and testing datasets, is mirrored by Random Forest and Gradient Boosting, which is a tendency of tree models. Note that Random Forest and Gradient Boosting achieve better performance than Decision Trees, reinforcing previous findings that these ensemble learning methods may generalize better and achieve higher performance than Decision Trees (Pranckevičius and Marcinkevičius, 2017).

In fact, all our models are overfitting on the training dataset, even with regularization techniques. We have already implemented TF-IDF, which reflects the importance of a word in a document within our dataset and assigns high TF-IDF scores to more relevant, informative words. However, the results for SVD and PCA may suggest that these dimensionality reduction techniques are not the most effective. Please see our Future Work section for our proposed improvements.

As a note on what metrics researchers should focus on: the prioritization of high precision or high recall depends on the specific requirements for the system. It seems reasonable to suggest that for a system leveraging ML models for disaster-related tweet classification that *high precision* should be the prioritized metric, since false alarms can be extremely costly. Given that resources may be limited during disaster relief, we want classifiers that minimize the number of false alarms to ensure that a timely allocation of resources and timely response can occur when real disasters occur. Moreover, given the nuanced aspects of language online, models should err on the side of caution and learn to not overflag tweets that may in reality only contain metaphorical or sarcastic language.

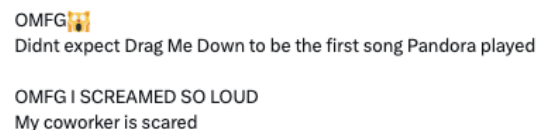
Finally, these classifiers should not sit in an infrastructure vacuum—existing instruments that predict and identify disasters (such as seismometers for earthquakes, buoy-based sensors or tide gauges for tsunamis, water level sensors for floods, satellite imagery, and so on) should still provide the primary source of information.

7 Limitations

We used a training dataset provided on Kaggle for the competition, Natural Language Processing with Disaster Tweets. While the competition provides over 10,000 tweets through a training and a testing dataset, the testing dataset ($n = 3,263$) was unlabeled. Given the volume of tweets and the relatively short project timeline, we chose to prioritize the development of machine learning models, as opposed to personally going through all 3,263 tweets and self-labeling.

Moreover, upon inspection of the training and testing datasets, both seemed to include several duplicates of certain tweets, which may have been due to retweets from different users. Having multiple duplicate instances may result in a model that is biased toward the content of these tweets, leading to a skewed understanding of the relationship between words or tokens in the dataset. Furthermore, some of the text that appears several times have different labels, indicating that a number of tweets are mislabeled.

Perhaps the most concerning limitation is that upon manual inspection, there are a number of dubiously labeled “disaster” tweets. For example, the tweet with identifier 8,450 is classified as a disaster-related tweet:



OMFG 🤯
Didnt expect Drag Me Down to be the first song Pandora played

OMFG I SCREAMED SO LOUD
My coworker is scared

While it may be more straightforward for a human reader to understand that the user is talking about their surprise and excitement that “Drag Me Down” by the boyband One Direction was played on their Pandora playlist, the last two lines may resemble real fear conveyed during a disaster. A number of these dubiously labeled tweets suggest that the dataset may have noise, and thus lead to models not learning what truly constitutes a disaster tweet.

Another risk may be that this dataset was created in 2019. Language is frequently evolving, so the definitions of words and slang may have changed over time. Thus, our predictions will be trained and based on older tweets from at least four or five years ago, which may no longer be the most accurate for current predictions.

In the next section, we propose how these limitations may be addressed through future work.

8 Future Work

To address the aforementioned issues with our dataset, we propose that a comparative analysis be conducted on a larger dataset with disaster-related tweets, particularly that is cleaned to have fewer repeats. Conducting analyses on more datasets can help us better understand whether plateaus in model performance are dataset-specific, or whether they are limitations of the models themselves.

To address overfitting, future work may perform feature selection through methods such as a Chi-Squared test and Pointwise Mutual Information (PMI). Feature selection eliminates irrelevant or redundant features, which would make our models more robust toward noise and overfitting to particularities of the training dataset. While our research focuses on model performance and comparing the performance between conventional models and deep neural network-based models, future work may also focus on feature importance selection for models and visualize the tokens or words that are most important in determining whether a tweet is disaster-related or not.

For the conventional models, future work may involve implementing additional hyperparameter tuning and improvements to the models. Given our limited computational power (T4 GPU provided by Google Colab), we were unable to run many parameters for grid search, particularly with Gradient Boosting, before the Colab Notebook would time-out. Because Kumar et al. (2019) found that Gradient Boosting achieved the highest performance, we believe that our inability to conduct a robust grid search may have inhibited our model performance.

For the LSTMs, we propose continuing to iteratively adjust and improve the architecture, focusing on reducing validation loss. Though we were unable to find a code repository for Kumar et al. (2019), examining other published LSTM architectures for text classification may serve as a point of reference for a better-designed model.

We used Gensim's GloVe-Twitter-200 word embeddings given the nature of the task (classification) and dataset (tweets). In addition to GloVe embeddings, Kumar et al. (2019) also use Crisis embeddings—domain-specific, disaster-related word representations developed by Artificial Intelligence for Digital Response (AIDR) (Alam, 2017). Doing so may reveal whether using domain-specific word embeddings improve model performance. Alternatively, given a sufficiently large

dataset, future work could also involve creating embeddings using a portion of the dataset to learn word representations that are most similar to the task at hand.

Finally, future work should also examine other neural network-based models, including GRUs and combination models, such as the GRU-CNN proposed in Kumar et al. (2019). The Kaggle dataset also proposes that competitors explore BERT and other transformer-based models, which appeared to achieve high performance on the leaderboard.

9 Conclusion

In this work, we implemented six conventional classifiers and two deep neural network models (bidirectional LSTMs) and compare their performance on a Kaggle disaster-related tweet dataset. We situate our work alongside (Kumar et al., 2019), which conducts a similar comparative analysis of traditional machine learning classifiers and deep neural network-based models. We find that out of the conventional models, Logistic Regression and Naive Bayes achieve the best performance as shown by their high precision, recall, and/or F1 scores. We also find that our improved LSTM achieves the best performance on the testing dataset out of all eight models, suggesting deep neural network models' ability to learn more appropriate, generalizable relationships in our text dataset.

Ethics Statement

This paper intends to contribute to a better understanding of conventional and recurrent neural network models, specifically with an application to disaster-related tweet classification. While this work is set against a backdrop of disaster relief agencies taking interest in using machine learning for faster responses to disasters, we do not intend for our work to be directly transferable to real-world deployment. This research focuses on the technical aspects of classification, selecting a meaningful domain of application, and is not produced with expert consideration regarding disaster relief.

Acknowledgements

We first thank our course instructors Dr. Gabriele Dragotto and Dr. Jake Snell for their instruction and guidance this semester. Thank you to graduate student preceptor Richard Zhu for providing suggestions on my project. Thank you to the COS484: Natural Language Processing staff for teaching me

about NLP techniques and inspiring me to work on this project.

Thank you to my friends and peers Lacey and Dyanne for filling SML310 with such a fun and positive energy. A very special thank you to my biggest supporter, Jeremy, for your love and encouragement always.

References

Firoj Alam. 2017. [Crisis Embedding Models](#). GitHub repository.

GeeksforGeeks. 2022. [Snowball Stemmer in NLP](#).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. [Long Short-Term Memory](#). *Neural Computation*, 9(8):1735–1780.

Addison Howard, Devvret Rishi, Phil Culliton, and Yufeng Guo. 2019. [Natural Language Processing with Disaster Tweets](#). Kaggle.

Abhinav Kumar, Jyoti Prakash Singh, and Sunil Saumya. 2019. [A Comparative Analysis of Machine Learning Techniques for Disaster-Related Tweet Classification](#).

Lori McNee. 2018. [The sky was ablaze over mount moran tonight](#). Twitter.

Jakub Nowak, Ahmet Taspinar, and Rafał Scherer. 2017. [Lstm Recurrent Neural Networks for Short Text and Sentiment Classification](#).

B. E. Parilla-Ferrer, P. L. Fernandez, and J. T. Ballena. 2014. [Automatic Classification of Disaster-Related Tweets](#).

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. [GloVe: Global vectors for word representation](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Tomas Pranckevičius and Virginijus Marcinkevičius. 2017. [Comparison of Naïve Bayes, Random Forest, Decision tree, Support Vector Machines, and Logistic Regression Classifiers for Text Reviews Classification](#). *Baltic Journal of Modern Computing*, 5(2):221–232.

Twitter. 2022. [When natural disasters happen, twitter can help. here’s how](#). Accessed: February 10, 2024.

A Appendix

A.1 Data

As a part of our exploratory data analysis, we visualized the 15 keywords with the highest frequencies for disaster-related tweets. We include the words and their frequencies as shown here.

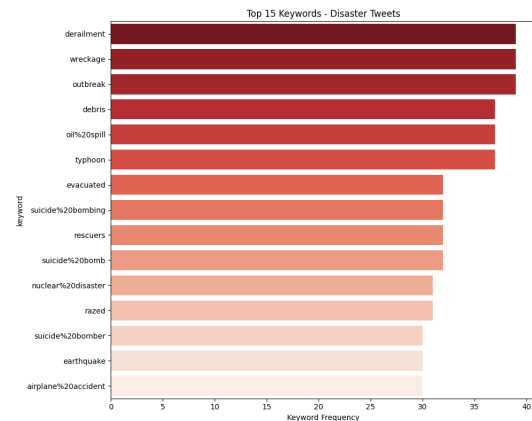


Figure 8: Top 15 Keywords (Disaster-related tweets)

We also include an example of a tweet that would potentially be interpreted by a classifier as a disaster-related tweet. Here, the user talks about the sky being “ablaze” over Mount Moran in Grand Teton National Park (McNee, 2018). While a human reader may easily understand the tweet to be discussing a vibrant sunset, a classifier might interpret the tweet as a concerning tweet about a wildfire in a National Park.



Figure 9: A tweet with metaphorical language (using the keyword “ablaze”)

A.2 Bag of Words

Below, we present the results of the six conventional models using a Bag of Words representation.

TABLE 4
Results for Six Conventional ML Classifiers on Disaster Tweet Dataset (Bag of Words)

		Training			Validation			Testing		
Model	Class	<i>P</i>	<i>R</i>	<i>F1</i>	<i>P</i>	<i>R</i>	<i>F1</i>	<i>P</i>	<i>R</i>	<i>F1</i>
Logistic Regression	0	0.88	0.95	0.91	0.78	0.85	0.81	0.80	0.85	0.83
	1	0.92	0.83	0.87	0.76	0.67	0.72	0.78	0.71	0.75
Multinomial Naive Bayes	0	0.84	0.91	0.87	0.80	0.86	0.83	0.79	0.86	0.83
	1	0.87	0.76	0.81	0.78	0.71	0.74	0.79	0.70	0.74
SVM (with RBF Kernel)	0	0.90	0.98	0.94	0.78	0.90	0.83	0.78	0.89	0.83
	1	0.97	0.86	0.91	0.83	0.65	0.73	0.82	0.66	0.73
Decision Trees	0	0.97	1.00	0.99	0.78	0.78	0.78	0.73	0.71	0.72
	1	1.00	0.97	0.98	0.70	0.70	0.70	0.62	0.64	0.63
Random Forest	0	0.98	0.99	0.99	0.79	0.81	0.80	0.77	0.78	0.77
	1	0.99	0.97	0.98	0.73	0.71	0.72	0.70	0.68	0.69
Gradient Boosting	0	0.90	0.99	0.94	0.77	0.88	0.82	0.77	0.85	0.81
	1	0.98	0.85	0.91	0.79	0.64	0.71	0.77	0.66	0.71