Analysis and Prediction of Tweets for Disaster Identification

Group P26:

Kenil Shah (kshah9)

Urvish Vasani (uvasani),

Chintan Gandhi (cagandhi)

Shahil Shah (sshah29)

Shahil Shah

Introduction and Related work

Introduction

- People are actively using social media platforms to announce emergencies in real-time. Twitter is one such platform.
- Tweets are monitored by news agencies and disaster relief organizations.
- Need mechanism to segregate real and fake disaster tweets so that timely action can be taken to address the disaster.
- Goal is to **predict whether tweet talks about a real disaster or not.**

Related Work

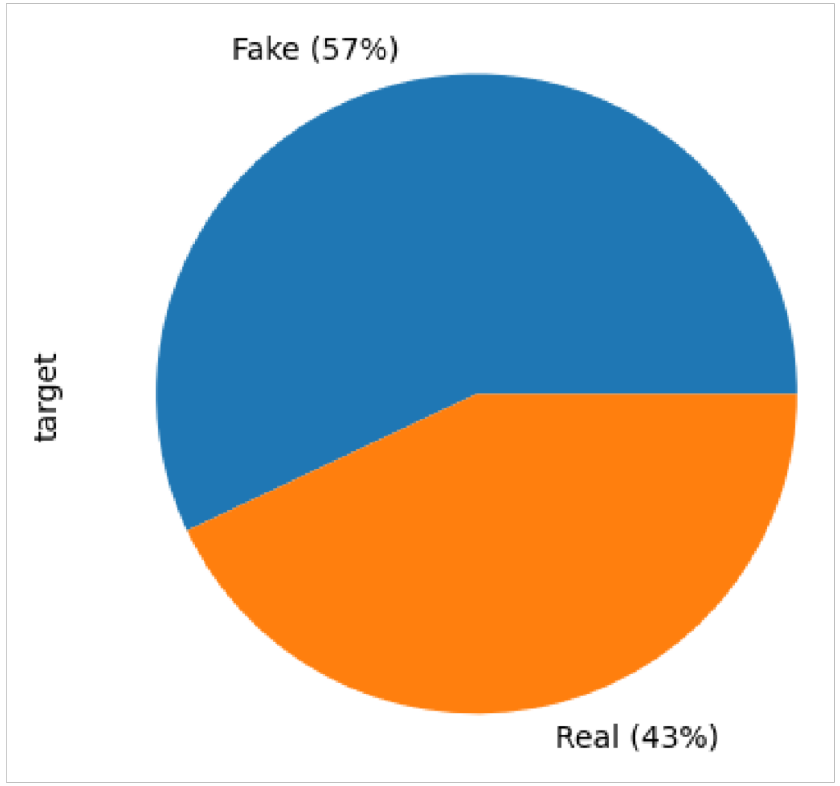
- An approach used by Zhang et al. [8] uses Word2Vec [9] approach for obtaining text embeddings followed by a Convolutional Neural Network and a Softmax binary classifier to categorize sentences.
- A semi-supervised sequence learning algorithm was proposed by Andrew et al. [6] that uses autoencoders as first step to finetune the parameters of the model and leverages LSTMs finetuned model to classify text.
- For vector representations of words people have used GloVe for various tasks that involves working with text.
- An unsupervised feature generating algorithm known as BERT [7] which was proposed by Google utilizes the strength of transfer learning and multi-layer bidirectional transformer encoder model to generate the vector embeddings for words.

*Citations are given at the end of the presentation.

Method

1. Dataset

- Total of 7613 tweets for training
- Total of 3263 tweets for testing
- Attributes of each tweet
  - Tweet from the user
  - Keyword from the tweet describing the disaster type
  - Location from where the tweet was sent
  - Target class
- Missing values in training data
  - Keyword: 0.8%
  - Location: 33%



Class Distribution of training tweets

2. Data Augmentation

- In order to avoid issues of overfitting, while training models with higher number of parameters we decided to use textual augmentation.
- Our augmentation strategy involved replacing certain words with their synonyms and add the new generated tweet to the dataset.
- For replacing certain words in the tweet we used GloVe as well as Wordnet architecture present in nlpaug library.

	Example 1	Example2
Original Tweet	I was in a horrible car accident this past Sunday	Usama bin Ladins family dead in airplane crash. Naturally no accident
Tweet1 using Glove	I was in a sick car accident this past sunday	Usama bin Ladins kinsperson dead in airplane crash. Naturally no fortuinity
Tweet2 using Glove	I was in a horrific car accident this Sunday	Usama bin Ladins family dead out landed crash . Naturally no accident
Tweet3 using Wordnet	I was in a atrocious car accident this past Sunday	Usama bin Ladins day dead in engine crash . Naturally hay accident

3. Approach

- **Preprocessing text data** :
  - Converting text to lowercase.
  - Removal of mentions i.e. words starting with '@'.
  - Removal of URLs and Hyperlinks.
  - Removal of punctuations.
  - Lemmatization.

1. **Original text:** Battle of the GOATS!!! https://t.co/sdjfv <a href=img.asp>
2. **Lowercase:** battle of the goats!!! https://t.co/sdjfv <a href=img.asp>
3. **Removing URL:** battle of the goats!!! <a href=img.asp>
4. **Removing HTML:** battle of goats!!!
5. **Removing Punctuations:** battle of the goats
6. **Lemmatization:** battle of the goat

- **Converting text to vector representation** :
  - Bag of word modelling (Count Vectorization)
  - TF-IDF vectorization
  - N-gram modelling
  - GloVe embeddings

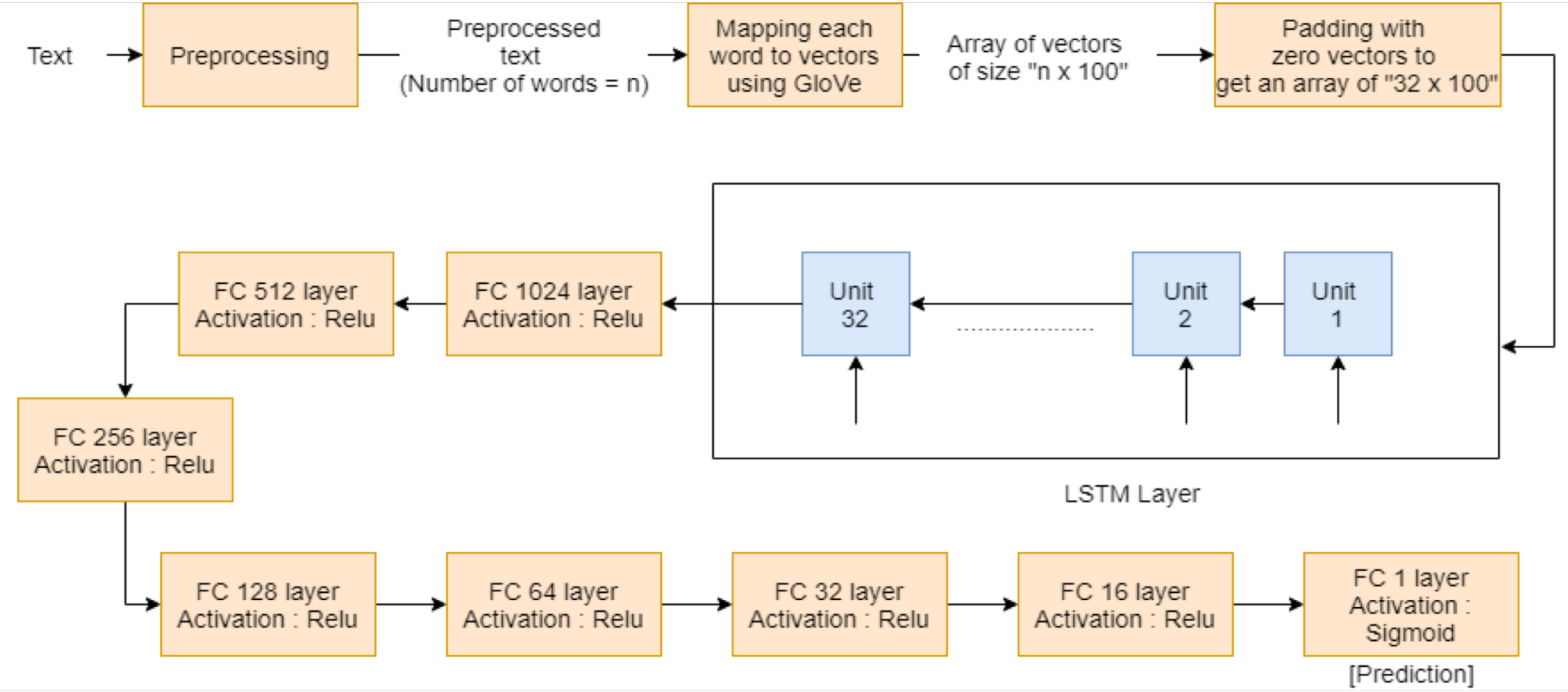
- **Classification models:**
  - Traditional ML models like Logistic Regression, Linear SVM, Multinomial Naïve Bayes, Decision Trees and Random Forest.
  - LSTM network with trainable embeddings and GloVe embeddings.
  - GRU network with GloVe embeddings.
  - BERT (Bidirectional Encoder Representations from Transformers) architecture.

Proposed Approach : GloVe + LSTM

- Data preprocessing as mentioned in previous slides.
- Converting each word to a vector representation using pre-trained GloVe embeddings on Twitter data.
- Padding the 2 dimensional array obtained from embedding layer for converting every sentence into a length of fixed size (32).
- Training a network with LSTM cells which is followed by a set of fully connected in order to classify a given tweet.

Kenil Shah

Final Model Architecture: GloVe Embedding with LSTM



Experiments

1. **Using CountVec and Traditional ML Algorithms:**
   – Used simple count vectorization to form a vector and trained various models: Logistic regression, Linear SVM, Multinomial Naïve Bayes, Decision Tree and Random forest.
2. **Using TF-IDF and Traditional ML Algorithms**
   – Performed TF-IDF vectorization to consider the relevancy of the word in classification. Similar set of models as before were trained using these vectors and parameters were tuned for best models.
3. **Using TF-IDF with N-gram modeling and Traditional ML Algorithms**
   – To consider the combination of adjacent words and utilize this partial sequential information, we have combined N-gram modeling along with TF-IDF vectorizer and trained same set of models as described before.

4. **Using trainable embedding layer and LSTM**
   – To consider long term dependencies between words and generate vectors more intelligently, we tried a LSTM based model with trainable embedding layer.
5. **Using pre-trained GloVe embedding and LSTM (Final Approach)**
   – In order to incorporate similarity between words and overcome issue of lack of training data for training embedding layer, we used pretrained GloVe embedding with two different models: wiki and twitter. Preprocessed & Augmented tweet data and used LSTM network to sequentially process each words in a sentence.
6. **Using pre-trained GloVe embedding layer and GRU**
   – We replaced LSTM cells from previous approach with another variant of Recurrent neural network, GRU (Gated recurrent unit).

7. **Weighted probability scores**
   - Leveraged the power of ensembling by combining the LSTM model with GloVe embedding to the probability scores derived from keywords present from the dataset.

8. **BERT (Bidirectional Encoder Representation of transformer)**
   - It is pre-trained on large dataset and fine-tuned for various tasked to generate contextualized word embeddings. We added a classification layer on top of the pretrained BERT architecture.

Results

Results

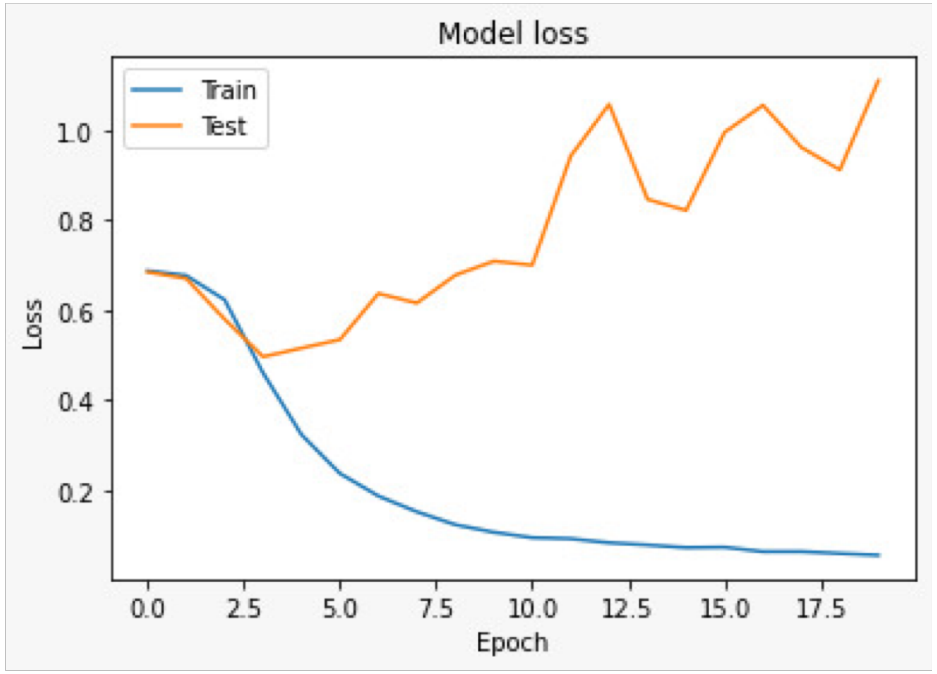
- Traditional ML algorithms:

Vectorization technique	Model with Best performance	Validation data Accuracy
CountVec	Multinomial Naïve Bayes	0.7597
TF-IDF	Multinomial Naïve Bayes	0.7449
TF-IDF with N-gram	Logistic Regression	0.7376

- After tuning the hyperparameters of these models with best performance, we received a best mean F1 score of 0.79 on the Kaggle challenge for Multinomial Naïve Bayes with TF-IDF vectorizer and Laplace smoothing parameter of 1.

- Deep Learning Algorithms

1. **Using trainable embedding layer and LSTM**: upon observing train/test loss curve, we found that possibly because of the insufficient amount of data for training both embedding layer and LSTM model led to issue of overfitting.

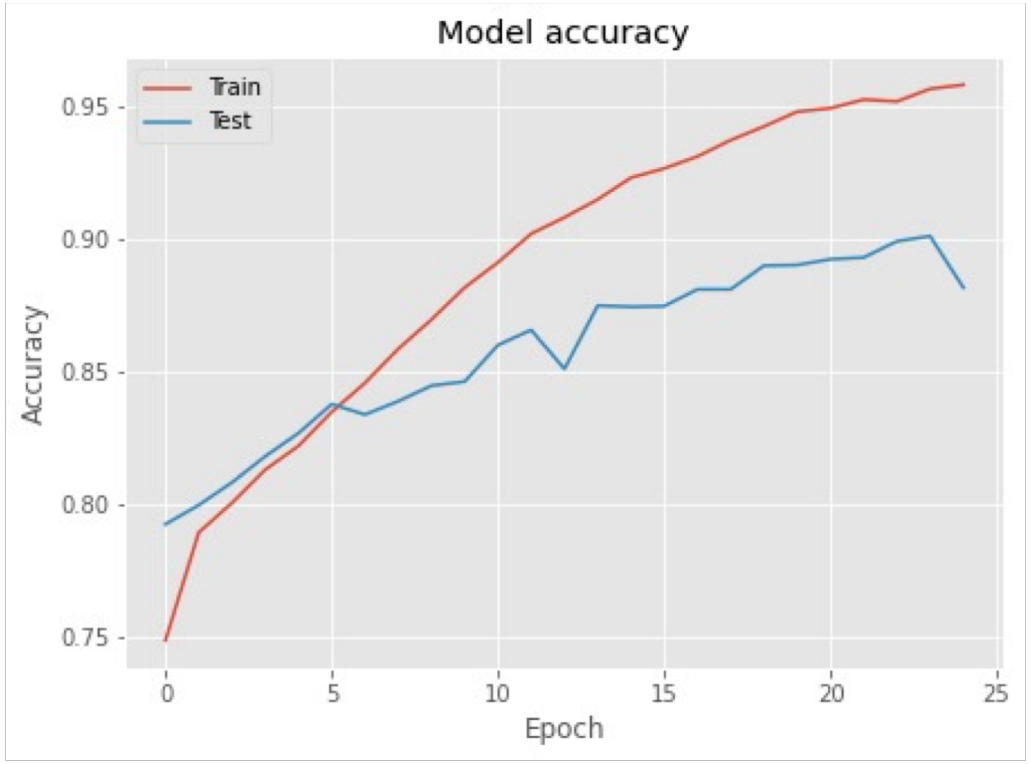


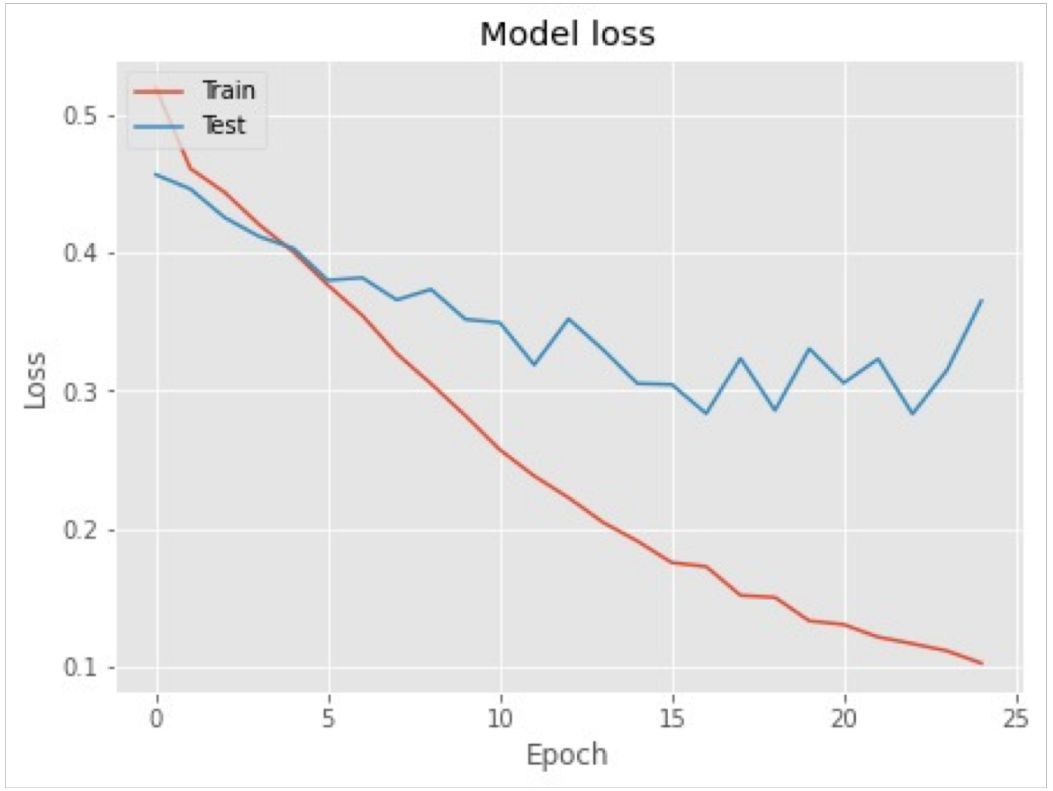
2. Using pre-trained GloVe embedding and LSTM:

- Two models were used for embedding and generating vectors. Embedding model with twitter data resulted in mean F1 score of 0.832 on Kaggle challenge whereas same for the model with wiki data was 0.817. In order to achieve these results, hyperparameters were tunes. These are the parameters that showed best performance:

    - Batch size = 64
    - Number of Epochs = 25
    - Dropout Rate for Fully connected layers = 0.2
    - Learning rate = 0.01
    - Optimizer = Adam
    - Loss Function = Binary Cross Entropy

Accuracy and Loss curves for final models





3. Using GloVe embedding layer and GRU:

- This approach didn't report better performance and received a validation accuracy of 77.2% and a mean F1 score of 0.6732 on Kaggle challenge.

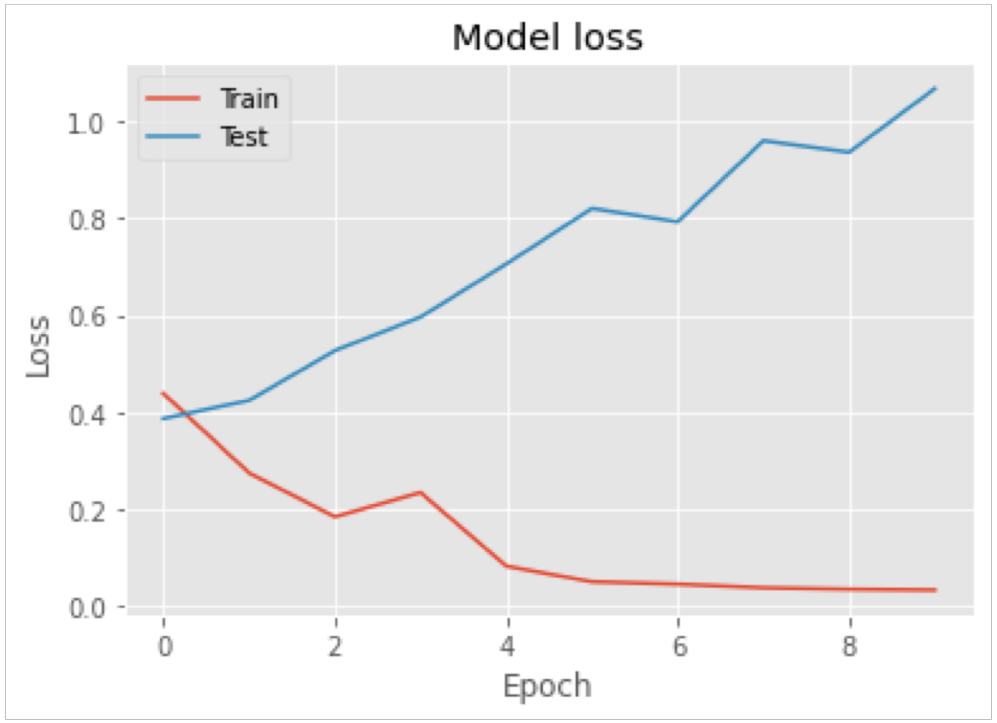
4. Weighted Probability Scores:

- Using weighted probabilities, following changes were observed for predictions on the test dataset:
  - Number of tweets whose class changed from real to fake: 98
  - Number of tweets whose class changed from fake to real: 117

Achieved mean F1 score of 0.830 on Kaggle challenge using this approach.

5. BERT (Bidirectional Encoder Representation of Transformers)

Figure given below describes the loss of model over epochs. Obtained mean F1 score of 0.826 on Kaggle challenge.



Comparison of Deep Learning approaches.

Table given below summarizes the best performance of each deep learning models. It can be observed that LSTM with GloVe embedding on Wiki data outperformed other approaches.

Architecture	Mean F1 scores on Test data
Glove(Twitter data) + LSTM	0.832
Glove(Wiki data) + LSTM	0.817
Glove Embedding with GRU	0.671
Weighted Probability Score	0.830
BERT + 2FC layers	0.826

Classification report for Final approach

Since we did not have access to the labels of testing dataset, we have used 20% of the training data as validation set to obtain classification report of our final GloVe Embedding (Twitter data) + LSTM model. Classification report is as below.

	Precision	Recall	F1-score	Support
0	0.85	0.95	0.90	2623
1	0.93	0.77	0.84	1945
Accuracy			0.88	4568
Macro avg	0.89	0.86	0.67	4568
Weighted avg	0.78	0.77	0.77	4568

Discussions

- **Why Traditional ML algorithms are not capable of achieving good results for this data?**
  - Traditional algorithms are not capable of encoding complete sequential information since it works with the vector representation of sentences. N-gram modeling can only encode some partial sequential information.

- **Why pretrained model with twitter data worked better than the wiki data?**
  - Structure of language and words aligns strongly with GloVe embeddings from Twitter data.

- **Why a lesser complex model (LSTM) performs better than a much more complex model (BERT)?**
  - Less size of training dataset results in complex model suffering from over-fitting.

References

- [1] Pennington, Jeffrey and Socher, Richard and Manning, Christopher D. 2014. Glove: Global vectors for word representation. Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 1532--1543.
- [2] Kaggle. Ongoing. Real or Not? NLP with Disaster Tweets. https://www.kaggle.com/c/nlp-getting-started.
- [3] Stanford NLP. 2020. GloVe. https://github.com/stanfordnlp/GloVe.
- [4] Hochreiter, Sepp and Schmidhuber, Jürgen. 1997. Long short-term memory. Neural Computation 9, 8 (1997), 1735-1780.
- [5] Kyunghyun Cho, Bart Van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078 (2014).*

- [6] Andrew M Dai and Quic V Le. 2015. Semi-supervised sequence learning. In *Advances in neural information processing systems. 3079-3087.*
- [7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805 (2018)*.
- [8] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).
- [9] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111-3119.

THANK YOU