Fake News Detection with ML

Team: Srushti Nandu and Mona Gandhi
Project Mentor TA: Ajay Patel

Abstract

With so much incoming news at once, and instant access to all kinds of news because of social media and other new media technologies, it becomes important to have a filter between the truth and trick. News is no longer restricted to reporting plain news. It is filled with biases, and other factors that tend to have a profound effect on our beliefs. One should always be aware about whether what he/she is reading has a legitimate source and whether it is reliable enough to base opinions on.

Github Link to Code: https://github.com/srushtinandu/ML-Project

Introduction

Fake news detection is an important task in this world, where we have access to social media, the internet and so many other sources of data. Verification of that data is important to be a more aware citizen of the world. Both of us are learning NLP right now, and this topic seems to interest us. We want to be able to detect fake news from a dataset that has already labeled news headings, via neural nets, namely transformers and LSTMs.

Related Prior Work

Early detection of fake news on Social Media: Yang Liu et al et al [5] propose a novel method for the early detection of fake news through classifying news propagation paths. They model the path of each news story as a multivariate time series in which each tuple is a numerical vector representing the characteristics of a user who engaged in spreading the news. They then build a time series classifier that incorporates both recurrent and convolutional networks which capture the global and local variants of user characteristics along the propagation path respectively, to detect fake news. Experimental results on three real-world datasets demonstrate that our proposed model can detect fake news with accuracy 85% and 92% on Twitter and Sina Weibo respectively in 5 minutes after it starts to spread.

Unsupervised rumor detection based on users' behaviors using neural networks: Chen et al [7] have proposed an unsupervised learning model combining recurrent neural networks and auto-encoders to distinguish rumors as anomalies from other credible micro-blogs based on users' behaviors. They propose comment-based features to exploit crowd wisdom to help detect rumors. The experimental results show that their proposed model was able to achieve an accuracy of 92.49% with an F1 score of 89.16%.

"Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection: William Yang, in the paper, presents a new publicly available dataset, labeled dataset for fake news detection. They collected a decade-long, 12.8K manually labeled short statements in various contexts. They also designed a novel, hybrid convolutional neural network to integrate meta-data with text, and show that this hybrid approach can improve a text-only deep learning model.

Formal Problem Setup (T, E, P)

T (Task): For a sample point, namely a statement text, one point is accompanied with the news statement, context, speaker's information, party affiliation. We train a classifier to predict whether a piece of news is true or false.

E (Experimental Setup): We have a publicly available labelled dataset called the LIAR dataset that has approximately 12.8k short statements of news. The news is labelled as one of the 6 categories - true, mostly true, half true, barely true, false, pants on fire. Aside from the statement, we are given features like context, speaker's information, party affiliation. We use the LSTM model using the context and statement feature, followed by the basic BERT model which is also a deep learning approach(but involves more context), using the statements to train the classifier to predict whether the news is true or false.

P (Performance Metrics): We have used accuracy as the performance metric for this task, both in LSTM and BERT models.

Methods

We have a dataset with 6 labels. As 6 class classification is difficult, we went for a 2 class classification problem. We combine true, mostly true and half true labels as true and the remaining barely true, false and pants on fire labels as false.

Our first approach uses LSTM, as it is used for sequential data. For LSTMs, we first pre-process the data by making the text lower case to have uniformity. Then, we tokenize all the words to feed it to an LSTM. We use the feature of Statements present in the dataset for training the model. We also remove the rows where this feature is null. We fine tuned these hyperparameters to get the best results. As we used Keras Sequential model, we use the EarlyStopping method for the number of epochs, but the maximum number of epochs are 7. We train the model. To get better results we tried using the context feature, but the accuracy did not improve much. Since the results were not promising using LSTM, we try a different approach.

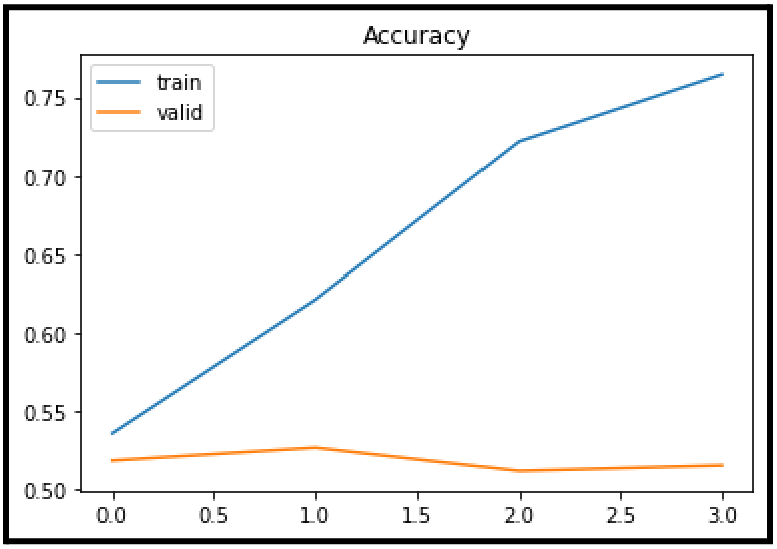
To overcome the shortcomings of LSTM, we switch to using BERT which is a pre-trained neural net approach. BERT stands for Bidirectional Encoding Representations for Transformers. It not only resolves the vanishing gradient issue as in LSTMs, but also captures the contexts of the words in the sentence, based on the words before and after it. Since it is pre-trained, we did not have to use any conversion for the words as well. We used the basic BERT model. The BERT model has some basic requirements for usage. Hence we had to do some data preprocessing before we could actually apply the model to the statements. It includes statements being of the same sentence size. So we padded the sentences, with the maximum length of the sentence being 100. We tokenized the sentences, changed words to ids and then fed that in the BERT model. We found it beneficial to use the cased model, because capitalization plays a major role in Named Entities, a major part of news.

Baseline approaches we compare against: Some other methods that we tried were Logistic regression, at the very beginning. The results were very hopeless as the statements and other features did not follow a particular trend. We also tried implementing naive bayes to our data, for classification. Because we used a bigram model, it gave us very low accuracy, worse than random assignment. We did not go further with any other baseline approaches because they had no way of capturing context and we predicted that it would not perform well.

Implementation Details: For the LSTM model, we tried different values for the probability for dropout, spatial dropout 1D and LSTM layer (it has recurrent dropout). We also tuned the size of the LSTM layer,

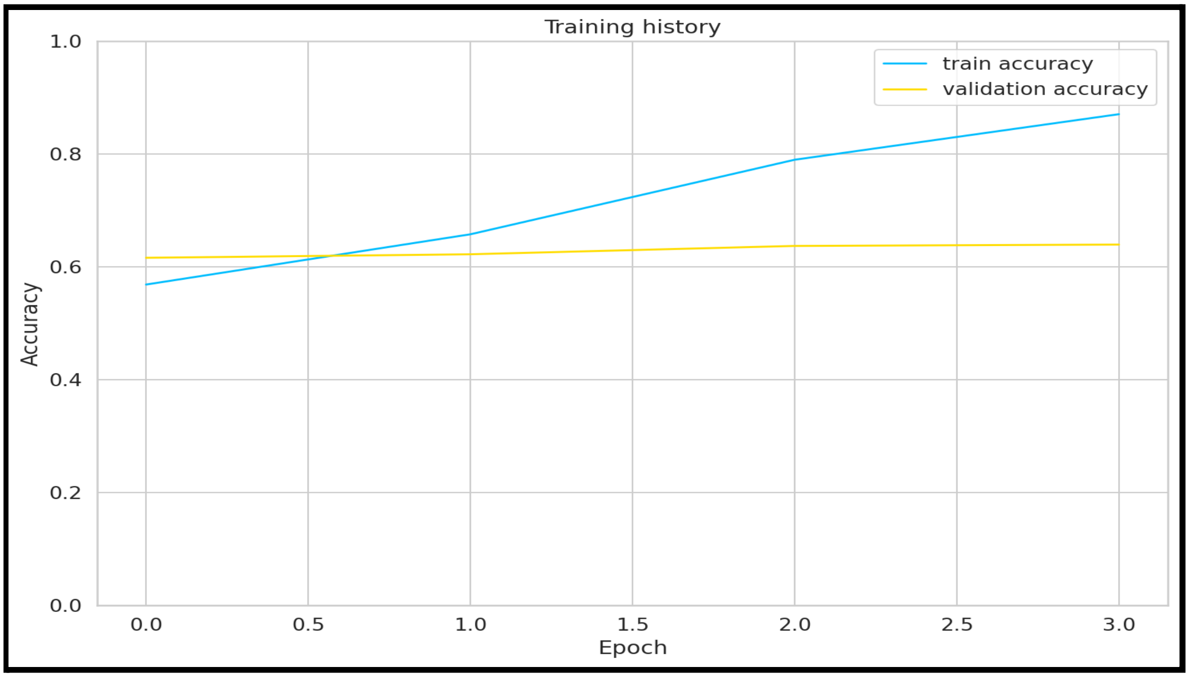
and the number of dense, dropout layers in the model. We first have an embedding layer, followed by a spatial dropout 1D layer whose output is used by an LSTM with 128 units, followed by a dense layer with relu activation, dropout layer (p=0.3) and another dense layer with relu activation.
Optimizer used: Adam
Loss function used: Binary Cross Entropy Loss.
The maximum number of epochs were 7, but the model stopped training after 4 epochs as the validation loss started increasing. We trained the model with a batch size of 32. The metric used is the binary accuracy. The Keras library helped a lot for hyperparameter tuning.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 50, 100)           1000000
_________________________________________________________________
spatial_dropout1d (SpatialDr (None, 50, 100)           0
_________________________________________________________________
lstm (LSTM)                  (None, 128)               117248
_________________________________________________________________
dense (Dense)                (None, 64)                8256
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 130
=================================================================
Total params: 1,125,634
Trainable params: 1,125,634
Non-trainable params: 0
```



Using the BERT model, we built our fake news classifier. We also use a dropout layer (p = 0.3) for some regularization and a fully-connected layer for our output.
Optimizer used: AdamW
Loss function used: Cross Entropy Loss.

We ran the model for 4 epochs, after which the model started overfitting.



Other Hyperparameters:
Learning rate = 2e-5
Batch size= 16
Epochs: 4

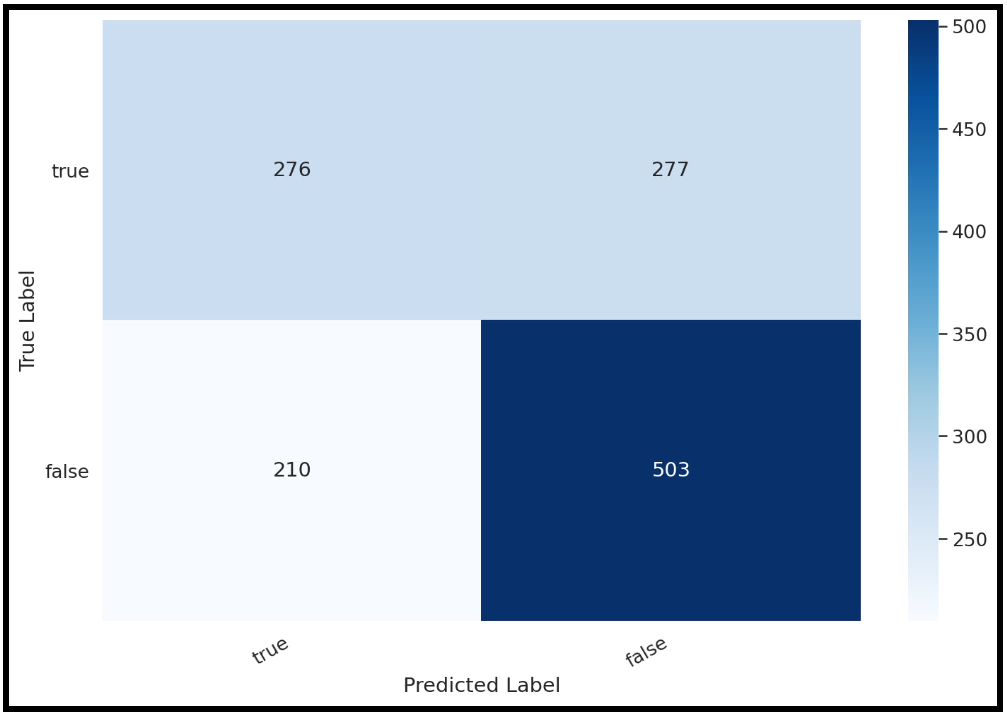
Experimental Results

We compare our BERT model with the LSTM model, against accuracy in the training, validation and testing data. We find that the BERT model works better. Below is the table that compares the accuracy of both the models on the three datasets.

Method Name	Training Data	Validation Data	Testing Data
LSTM model	0.7717	0.5156	0.534
BERT model	0.8702	0.6391	0.6153

Observe that the LSTM model(validation and testing) accuracy is not very high. It is just a little better than a random assignment of labels.
We see that the BERT model does better than the LSTM model in terms of accuracy. The validation accuracy seems to generalize well on test data, which means it is a good representation of the unseen test data. The accuracy is not enough to rely on. Hence, the conclusion could be that we need a better model or more data. It is both. In a diverse domain like news, even 10k statements can fall short to train the network. More data will definitely help the model, but overfitting issues may have to be combatted.

Next we study the confusion matrix, that will help us know better to know exactly what kind of samples our model does better on.



Observe that the model does better on false statements. It classifies the false statements better. It does a sloppy job with classifying true statements.

Conclusions and Future Work

Overall an accuracy of around 62% is not bad for this basic model. It shows that there is a lot of scope to improve accuracy for testing data by employing better methods. One of the future work possible is instead of classifying it into binary classes, we give out the probability using linear regression. This is also helpful to make the 2 class classification into 6 class classification. We also wish to combine two or more models to get better accuracy, like we could have BERT for statement and LSTM for context whose output layers are concatenated and used as input to another feed forward neural network. Another thing that could help would be looking at the statements in all four sections of the confusion matrix, and and try and see which features the model is using to classify. During the time of this pandemic, we also wish to use this fake new detector for covid news. It would help people distinguish between fake and genuine news.

Ethical Considerations and Broader Impacts:

News sways people’s thoughts and mindsets in different ways. You can control people’s opinions via presenting a certain piece of news, and presenting it in the way you want. We need to be very careful while making such systems, because correct classification is very important. We cannot go wrong if people are going to rely on the system to decide what party they want to vote for, whether or not any political news that has come out is false. These bring up some major ethical concerns in our way. We are also opening the model to incorporate biases based on training data, against a particular leader, party, etc. Our system needs to be made robust to biases before it can be used by the public.

Prior Work / References:

1. 1. William Yang Wang, “Liar, Liar Pants on Fire”: A New Benchmark Dataset for Fake News Detection, May 2017
2. 2. Ranjan Ekagra, Fake News Detection by Learning Convolution Filters through Contextualized Attention, August 2019
3. 3. bedarkarpriyanka/NLP-Project-Fake-News-Detection
4. 4. manideep2510/siamese-BERT-fake-news-detection-LIAR
5. 5. Yang Liu, Yi-Fang Wu, Early detection of fake news on Social Media through propagation path classification with recurrent and convolutional networks, April 2018

Supplementary material:

(Example: cool large image results that couldn’t fit in the main report etc. Only put non-essential bonus stuff here --- remember, graders might not see this)