Fake News Detection Using Deep Learning

Rohan Badal, Fardin Hussain, Bradley Kerr, Margaret Larkin, Anjali Sharma
University of Michigan, EECS Department
{rbadal, fardin, kerrb, malarkin, shanjali}@umich.edu

Abstract

Identifying fake news automatically poses a significant challenge, with far-reaching political and social consequences in the real world. In today's age of unencumbered access to the internet, there is an increased risk of misinformation. The boom of artificial intelligence also enables easy production and propagation of fake news. This issue is particularly prevelent in the political sphere as false information can swing elections and in turn affect millions of lives. Since it is difficult to manually determine the validity of digital media, a more effective solution may involve the use of automated machine learning systems. We propose a machine learning textual analysis tool to determine the validity of political statements based purely on language structure. This paper outlines the design and architecture of a fine-tuned DistilBERT model using text, metadata, and textual sentiment, improving upon locally trained models. This system could empower individuals to filter fake news out of their media, encouraging the general population to become more informed.

1 Introduction

Problem Statement: Digital media plays an increasingly significant role in the average person's consumption of news as the internet becomes more prevalent. An estimated 93% of Americans receive some of their news from a computer, smartphone, or tablet, with 56% receiving all of their news in this manner (Pew Research Center, 2023). With more of the population gaining access to this tool, the internet provides an anonymous platform to an unprecedented amount of people enabling easy production and propagation of fake news. Misinformation has particularly significant consequences for the political sphere; false information can swing elections and in turn affect millions of lives. Since it is difficult to manually determine the veracity of digital media, a more effective solution may involve the use of automated machine learning systems. We propose a machine learning textual analysis tool to determine the validity of political statements based purely on the text and its metadata. This system could empower individuals to filter fake news out of their media, enabling the general population to stay informed.

Related Work: Fake News Detection has become a widely-researched problem, and given the recent boom in deep learning applications of natural language processing, many have investigated a machine learning approach. Older datasets used for deep learning (Kaliyar, 2021) typically support only binary classification, which is not nuanced enough to capture the true label space, and oftentimes these datasets lack credible annotations. The LIAR dataset (Wang, 2017) has become a benchmark due to its size, multiclass labels, and certified annotations. Others such as (Aslam, 2021) have produced deep learning models that achieve 90% accuracy on this dataset, but include speaker credibility statistics in their features, which we believe introduces great bias and practicality concerns for any real life implementation. We experiment with replacing this incriminating data with self-defined sentiment tags in a multimodel system. Other approaches (Upadhayay, 2020) experiment with sentiment feature extraction, but include the bias-inducing speaker credibility features as well. The question still remains: how well can the most proficient deep learning methods classify the LIAR dataset while excluding the overly revealing metadata?

Solution: We answer this question by deploying two deep learning approaches: a convolutional neural network and a pretrained large language model, DistilBERT. Additionally, we experiment with generating sentiment tags for each sample and propose alternative evaluation metrics to standard accuracy. Our model takes text from a news snippet and op-

tional metadata as input and estimates the degree of misinformation. We are able to reproduce benchmark results for this dataset using such methods.

Outline: In this paper, we discuss the development of our software system and our experiments. First we will explain the dataset and feature extraction methods, followed by the model selection, design, and architecture for each of the two machine learning models we deploy. We follow with an evaluation of the results of our tests, and we finish with an exploration of the ethical concerns as well as the potential future work in this field.

2 The LIAR Dataset

We chose to use the LIAR dataset from Hugging Face, which was inspired by similar work in the field. More specifically, William Yang Wang conducted research on the LIAR data set within the Computer Science department at UCSB. He implemented a handful of locally trained models, the most successful being a Hybrid CNN trained on text and speaker data with an accuracy ranging from about 23-27 percent. We used this as our baseline accuracy.

The dataset contains statements and there corresponding classification, with the intention of training a model to determine truthfulness based on speech patterns. LIAR offers 6 labels: true, mostly-true, half-true, barely-true, false, and pants on fire, all sourced from politifact.com, which is a website dedicated to fact-checking. We are assuming that the classifications given in the dataset are correct and reliable. A justification and the editor that made the classification can be found on politifact.com. Although the multi-label setup makes it far more difficult for the model to predict correctly, we chose to use it anyways since it provides a more nuanced view of truthfulness. Most statements have partially true or false components, which this labeling scheme acknowledges.

Beyond a statement and classification, LIAR also included the following metadata: {subject, speaker, job_title, state_info, party_affiliation, barely_true_counts, false_counts, half_true_counts, mostly_true_counts, and pants_on_fire_counts} The counts corresponded to how frequently a speaker was noted for a specific classification. Interestingly, true_counts were not included, making the counts unnecessarily biased toward specific speakers. Regardless of the metadata including in training, the model was prone to overfitting. In the end we chose to include all textual metadata and only exclude numerical data since it provided the best results.

During the beginning stages of research. We used a decision tree classifier, trigrams, and text-length to gain insight into the text data and metadata. This gave us an idea of what trends we may be looking for.

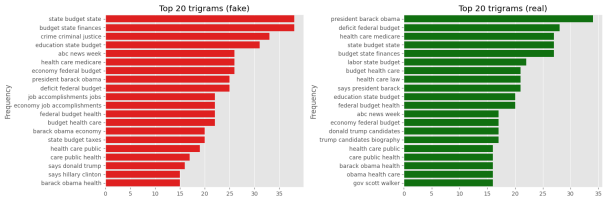

Figure 1: Top Trigrams

Figure 1 shows the top trigrams found in both fake and real data. There are many names present, which caused us worry that speaker data may encourage over-fitting. However, this issue was somewhat unavoidable given the names mentioned within the textual data.

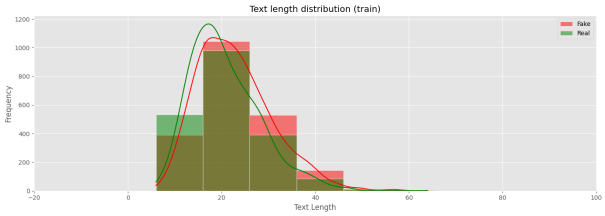

Figure 2: Text Lengths for Real and Fake data

Figure 2 shows the frequency of text lengths for real and fake data. Although this data was not included in our model, it is interesting to note that shorter statements were often classified as true, where as longer statements were more likely to be classified as false. This dataset is quite broad including subjects from healthcare, to elections, to the federal budget. It also spans a variety of sources from something as widespread as ABC news week to something as small as a town hall in Austin, Texas. For this reason, the model relies heavily on textual statements. These statements are shorter than others we have seen in datasets used for fake news detection, which gave us an opportunity to see if fake news can be classified accurately when relying so heavily on language structure and speech patterns.

3 Feature Extraction

To help protect against over-fitting, we aimed to diversify our feature set beyond metadata like speaker or party affiliation. The two features we looked into were Sentiment and Politeness. We utilized the Hugging Face pipeline to make these classifications, specifically the following models: {"sentiment-analysis", "NOVA-vision-language/polite_bert"}. Using the decision tree classifier mentioned earlier, we determined that Sentiment was an influential feature, which is supported in literature (Upadhayay, 2020). As for politeness, we consistently ran into issue where nearly every statement was classified as neutral. When omitting the neutral class, the feature was not important compared to the other available metadata, so we chose to remove it from training. In future iterations of this model, there are other processes that provide politeness features rather than a simply classifying a statement as polite or not polite. This could potentially be used to create a politeness index that could be a more useful metric in training.

During feature extraction, we also experimented with reducing the label space from 6 to 3 categories: likely-false, unsure, and likely true. The aim was to enable better label discrimination while maintaining a more complex understanding of truthfulness.

4 Model Architecture

Broadly speaking, we used three different model architectures to classify the news statements: CNN, CNN - RNN, and DistilBERT finetuning.

Based on previous research, it seemed that CNN-based architectures provide the best result for news classification when training the model locally. Hence, we implemented the CNN and CNN-RNN architectures to get a baseline result to compare to previous research. To obtain a potentially more robust model with better results, we decided to fine-tune a pretrained model, DistilBERT. DistilBERT (where BERT stands for Bidirectional Encoder Representations from Transformers) is rooted in the Transformer architecture, and is already trained on natural language. Due to the higher model complexity of the DistilBERT model alongside the fact that it has already been trained on natural language made us optimistic that the DistilBERT architecture could provide us better results (which, it actually did!).

4.1 CNN

The first model we implemented was a traditional CNN model, with the statement data being tokenized using the Google News word2vec model. The embedding layer maps each word in the statements to a numerical representation using the Google News model. We use four one dimensional convolutional layers with 36 filters each and varying kernel sizes. The dropout layer is there to help prevent overfitting. Finally, the fully connected linear layer outputs a vector of probabilities for each label. This is illustrated by the CNN model layout in Figure 3.

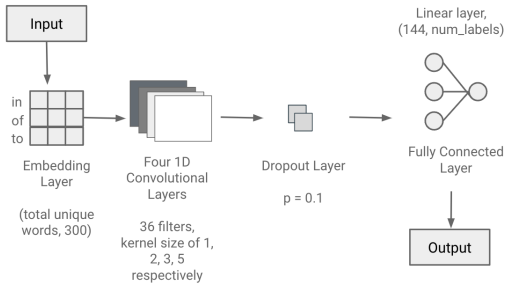

Figure 3: CNN Model

There were some issues with the initial model, namely that the Google News model does not account for all the unique words in the statement corpus, and that the relationship between words and context was being lost.

4.2 CNN-RNN

For this model, we implemented a CNN-RNN hybrid with two bidirectional LSTM layers. The convolutional layer helps to condense the input data down and to extract different features and patterns in the statements. The two bidirectional layers help to capture not only the context and connections between words, but also the past and future data points of the statements. The model layout is shown in Figure 4 below.

We process the data using a custom tokenizer in order to take into account all the possible unique words in the statement corpus. The embedding layer uses those tokenized words to map them to their vector representations. The data is then passed to a single one dimensional convolutional layer, with 128 filters and a kernel size of 5. We found that more convolutional layers made the performance worse for this particular model. Then

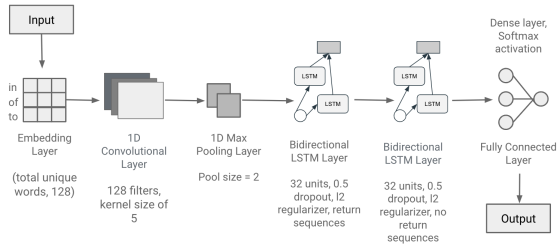

Figure 4: CNN-RNN Model

the max pooling layer with a pool size of 2 condenses the data down. Next, the two bidirectional LSTM layers capture the context and relationships between words. And, the fully connected dense layer outputs the predicted labels for each statement.

4.3 LLM – Finetuning DistilBERT

We finetuned the LLM on four different feature spaces:
(i) News Text
(ii) News Text and Textual Metadata
(iii) News Text, Textual Metadata and Sentiment
(iv) News Text, Textual Metadata, Sentiment, and Numerical Metadata

While the broad architecture of the model remains fairly constant regardless of the feature space used, each of the above four feature spaces require a slightly different variation to the LLM finetuning model.

Initially, we preprocess the raw news text by converting it into a format that DistilBERT can understand via tokenizing the text and converting the tokens to their corresponding IDs in the DistilBERT vocabulary (alongisde padding and truncating the text sequence to a fixed length). We do so via the DistilBERT tokenizer. For models (ii), (iii) and (iv) where textual metadata is used, we append the following sentence to the end of the news text, before passing it into the tokenizer:

"This is what {speaker name}, a {political party name}, said on {subject}."

Once the tokenized text is fed to the DistilBERT model, it produces a 768 dimensional output vector. Then, we pass our data through a (768 + x, 768 + x) fully connected layer, as noted in Figure 5. In models (i) and (ii), we have x = 0, since these models do not use the sentiment value or the numerical metadata. For model (iii), we add on a neuron that contains the sentiment value of the text (+1 for positive sentiment and 0 for negative sentiment) to the 768 neurons from DistilBERT and pass our data through a (769, 769) fully connected layer (giving us x = 1). For model (iv), we use five numerical metadata values (barely_true_counts, false_counts, half_true_counts, mostly_true_counts, and pants_on_fire_counts) alongside the sentiment, giving us x = 6.

The 768 + x output neurons from the first fully connected layer in the model is then passed into the Tanh activation function to introduce non-linearity in the model. After the activation function, we apply a dropout layer to reduce overfitting, which is crucial in this case due to the inherently high complexity of this model. Afterwards, we apply another fully connected layer of dimensions (768 + x, num_classes). For each of the four feature schemes noted above, we train the model with both three and six distinct classifications. Hence, num_classes is always equal to 3 or 6. To ensure the num_classes neurons in the output of this fully connected layer lie in the range of [0, 1] and act as probabilities for each of the classes, we apply softmax on the num_class neurons of the output layer.

Since the nature of our labels is ordinal and we want to reflect this fact when training the model, we convert the probability values for each of the labels into cumulative probabilities. This ensures that even when the model makes a "wrong" prediction during inference, the predicted class is not too far off from the actual class (i.e. better ordinal accuracy). We also convert the labels in the training dataset to a vector of cumulative probabilities for each of the labels (i.e. labels that are more false than target get zero value and the others get a value of 1) and use that when performing the training.

Also, to reflect the fact that the model predicts cumulative probabilities when training, we use a custom-made loss function, instead of the usual Cross-Entropy Loss. This custom-loss function computes the sum of the Binary-Cross-Entropy Loss for the cumulative probability value for each class and then divides it by the number of classes.

For all the eight models that we train, we obtained best results when training for two epochs, with a learning rate of 5e-5 in the first epoch and 1e-5 in the second epoch. The training-batch size used was 16 throughout.

When using the model to perform the inference on the given news text, we convert the computed cumulative probabilities into categorical probabilities

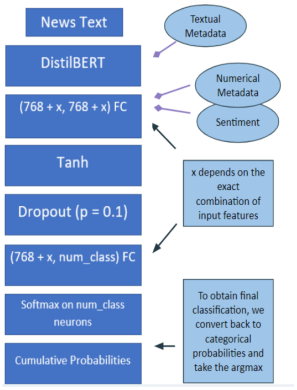

Figure 5: Neural Net for Finetuning DistilBERT

and output the class with the maximum categorical probability.

5 Model Evaluation

5.1 Ordinal Accuracy

Classification of truth, contrary to intuition, is not binary. Truth can be more of a spectrum, and our label space reflects that. Because this is the case, a standard accuracy measure is not always a good representation of how well the model performs. For example, if the ground truth is "completely true", and our model predicts "mostly true", a standard accuracy measure would give no credit for this answer. In reality, "mostly true" is still a much more satisfactory answer than something like "completely false". In order to represent this idea of closeness, we have incorporated ordinal accuracy. Ordinal accuracy gives partial credit to classifications that are close to the correct classification. In the case of this model, classifications within 1 label of the true label were given 0.5 credit.

5.2 Results

Our models perform equally well for both the cases when we use 3 distinct classifications for the news, and when we use 6 distinct classifications.

While the accuracies are numerically higher across the board when we use 3 labels as opposed to 6, it is important to keep in mind that the accuracy values have a higher threshold to be deemed as reasonable when using 3 labels as opposed to 6. For example, for our model to produce inferences better than a random guesser, we would need to surpass 0.333 accuracy when using 3 labels and 0.1667 accuracy when using 6 labels.

The usage of textual metadata on top of the text improved accuracy values, as expected. The addition of sentiments on top of the textual metadata gives us even better results for accuracies. However, when we add in the numerical metadata, the accuracy values go down a bit. Hence, the feature space of text, textual metadata and the sentiment produces the best results for the LLM-based model that finetunes DistilBERT.

Features	Labels	Val. Acc.	Test Acc.	Test Ordinal Acc.
Text	3	0.464	0.468	0.673
Text	6	0.276	0.285	0.455
Text, TM	3	0.487	0.500	0.689
Text, TM	6	0.297	0.296	0.475
Text, TM, S	3	0.496	0.499	0.698
Text, TM, S	6	0.286	0.301	0.486
Text, TM, S, NM	3	0.477	0.479	0.686
Text, TM, S, NM	6	0.285	0.291	0.469

Results for Finetuning DistilBERT Architectures

Key — TM=Textual Metadata, S=Sentiment, NM=Numerical Metadata

Model	Features	Labels	Test Acc.	Test Ordinal Acc.
CNN	Text	3	0.451	0.651
CNN	Text	6	0.242	0.429
CNN-RNN	Text, TM	3	0.443	0.668
CNN-RNN	Text, TM	6	0.239	0.416

Results for CNN-based Architectures

Key — TM=Textual Metadata

For CNN-based architectures, the addition of textual metadata with the CNN-RNN model did not improve the model's performance. Due to the nature of the LSTMs, it is likely that the textual metadata just served as noise leading to a decrease in accuracy.

Overall, we were able to meet the average CNN accuracy seen in William Yang Wang's study when using just the news text tokens as the only features (without any form of metadata) when training the CNN. When we finetuned the pre-trained DistilBERT model with just the text (without any form of metadata), we obtained objectively better results for both ordinal and standard accuracies on the testing set, regardless of the number of labels. When we finetuned the pre-trained DistilBERT on a feature space encompassing the text, textual metadata and the binary sentiment of the text, we obtained the best results for both the three-label and six-label models.

6 Ethical Implications

There are a few ethical implications of our model. First, we trained our models on the LIAR dataset under the assumption that its labelling of the statements is the ground truth. PolitiFact can have its own political biases, so there is a risk that the statement labels are not fully accurate.

Additionally, there is a high risk of overfitting based on a single feature like speaker or source. For example, if a certain speaker is associated with likely true statements frequently in the training set, their statements will be marked as likely true even if they are actually false. This is a serious problem that is difficult to rectify. We need to use more data with an even distribution of different speakers, sources, political affiliation, etc. We might also need to omit certain features in order to avoid any bias in the model towards a specific person or source.

Another ethical implication of our model is the potential for unfair classifications. For example, an unfair classification might be when a statement is marked as false when most of its information is almost entirely true. To fix this problem, we are using a dataset with six labels instead of just true or false. The six labels (true, mostly- true, half-true, barely-true, false, and pants on fire) offer more flexibility and accuracy for how the model labels the statements. The user also gets more information about statements they input from the model because of the variety of labels.

Lastly, users may misinterpret our model as ground truth instead of a tool for trying to assess if a statement is potentially misleading or not. In order to deal with this problem, we can include disclaimers on the final product stating that our model is not meant to be used as a final assessment of truthfulness, rather it is an indication that the user may need to look into their sources more carefully.

7 Limitations

Our model does have some limitations. First, while it does perform well with political statements, it does not generalize well with other types of data, such as pop culture references and social media posts. Training on a larger variety of data with more features may help to mitigate this issue. Sentence categorization can also help to make sure that the model is not classifying non-declarative sentences or clear opinions as potentially true or misleading.

Additionally, we do not incorporate ground truth into our model, which requires significantly more robust training that we do not have the computational resources for at this time. If we had more resources, we could potentially incorporate Retrieval Augmented Generation (RAG) to run searches to fact check statements. We could also look into how emotion can impact a speaker's veracity by exploring more complex sentiment analysis, valence/arousal, and audio files.

Lastly, not all input data will have the metadata we trained our models on. This can result in incorrect label predictions, which can be a serious problem. Users can be misled if it labels true information false, or vice versa. As a result, that misinformation can spread, causing even more damage and harm. More robust training and model options can be implemented to help minimize this risk. Additionally, having users input the data and metadata of their choosing can help with better model selection. For instance, if they input a statement only, then our tool can select the model best suited for just statement analysis. Ultimately, more computational resources are required to deal with these limitations.

8 Conclusion

Through our experiments with deep learning approaches to classifying fake news, we reveal some insights. While we replicated benchmark results for fake news classification on LIAR without metadata, the models we produced are far from reliable. DistilBERT outperformed the CNN in all categories, but their accuracy scores are still significantly lower than one would need to rely on the models' predictions. Fake news detection is a very difficult task for a machine learning model based purely on textual qualities, as oftentimes they are intended to be deceiving. However, more work needs to be conducted on this problem before these approaches can be ruled out. Particularly, retrieval augmented generation (Soman, 2024) allows for promising applications for informing LLM's with contemporary factual information. Additonally, more efforts should be made to reach a consensus about which metadata is appropriate to include. We experimented with using no metadata and using only textual metadata, but there are arguments to be made for including the speaker credibility data. This tradeoff between inputting more metadata and the risk of introducing more biases should be explored in a full-fledged study.

9 References

B. Upadhayay, V. Behzadan, *Sentimental LIAR: Extended Corpus and Deep Learning Models for Fake Claim Classification*, 2020 IEEE International Conference on Intelligence and Security Informatics (ISI), Arlington, VA, USA, 2020, pp. 1-6, doi: 10.1109/ISI49825.2020.9280528.

Kaliyar, R.K., Goswami, A. Narang, P. *FakeBERT: Fake news detection in social media with a BERT-based deep learning approach*. Multimed Tools Appl 80, 11765–11788 (2021). https://doi.org/10.1007/s11042-020-10183-2

News Platform Fact Sheet. Pew Research Center, Pew Research Center, 15 Nov. 2023, www.pewresearch.org/journalism/fact-sheet/news-platform-fact-sheet/.

Nida Aslam, Irfan Ullah Khan, Farah Salem Alotaibi, Lama Abdulaziz Aldaej, Asma Khaled Aldubaikil, *Fake Detect: A Deep Learning Ensemble Model for Fake News Detection*, Complexity, vol. 2021, Article ID 5557784, 8 pages, 2021.

Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H. (2017). *Fake news detection on social media: A data mining perspective*. ACM SIGKDD explorations newsletter, 19(1), 22-36.

Soman, S., Roychowdhury, S. (2024). *Observations on Building RAG Systems for Technical Documents*. arXiv preprint arXiv:2404.00657.

William Yang Wang. 2017. *Liar, Liar Pants on Fire: A New Benchmark Dataset for Fake News Detection*. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 422–426, Vancouver, Canada. Association for Computational Linguistics.

10 Appendix

Appendix A. Github Repository

https://github.com/TooPercentMilk/PersonalityHires