# Deception Detection in Tweets: Neural network models for credibility classification

**Isabel Corpus and Gina Markov**

## 1 Introduction

A common concern in text classification tasks is that of deception detection. This task of classifying documents as fraudulent or legitimate is represented in spam detection efforts, fake news detection, and image recognition. The task of identifying 'fake news' has become an increasingly important question as government bodies and other institutions seek to mitigate the prevalence of fake news in contributing to political extremism. The pernicious role social media plays in the spread of fake news became apparent during the 2016 U.S. election. Despite past research that has been done investigating effective methods of identifying news articles that are fraudulent, there has been less attention paid to the case of untruthful social media content. For our final project, we adopted an LSTM and several BERT language models as well as more traditional classification models, such as SVM, Naïve Bayes, and k-NN to classify political tweets as fake or trusted. We chose to run these more traditional methods so we could hand select the features that would be used in the classification decision. This gave us insight into BERT's performance.

This research filled a gap in the existing corpus of deception detection literature in several ways. First, we analyzed the Politifact LIAR corpus in a new way, by comparing our network accuracy with the performance of hand-coded models such as the SVM and Naïve Bayes models. Much of the literature regarding deception classification tasks have focused on news headlines or events, but we will instead be using claims pulled from social media. Second, there are a variety of contextual and semantic methods that have been used to determine the probability that a document is untrustworthy. We use a bag of words and bag of N-grams as input to our machine learning models.

## 2 Literature Review

Research regarding detecting fake news has oftentimes focused on news articles and blog posts. One such article by Da San Martino et al. argued in favor of a fine-grained analysis of news articles (Da San Martino et al., 2019). In this paper, the authors constructed a corpus of news articles and set of evaluation criteria for labeling news articles as legitimate or propaganda. The authors found that using additional information, such as news source or author, introduced noise that worsened model accuracy. They instead opted for network based architecture that used 18 propagandistic techniques. The techniques included vocabulary features, such as 'loaded language' that looked at emotional sentiment in the vocabulary. 'Exaggeration' was another example

of a vocabulary feature, where frequent use of superlatives was hypothesized to be correlated with higher likelihood of fraudulence. An example of another linguistic feature the authors identified to be a predictive feature of propaganda was "repetition", where the same phrases or messages were repeated over and over again over the body of the news article. After partnering with a firm that specializes in annotating data sets, the authors were able to gather counts of all instances of each technique and gather the parts of the sentence that were identified to contain "propagandistic elements". Using these phrases, Da San Martino et al. created a classifier to predict if a given sentence contained a propagandistic phrase. This classifier was composed of a BERT with three linear layers added to perform classification across the eighteen propagandistic tasks. Their focus on specific instances of phrases within the document suggested that a similar method could be applied to tweets and other shorter forms of news expression.

Another relevant article was written by Yale Professor Dragomir Radev in 2010 (Hassan et al., 2010). This article involved classifying the attitude of sentences. This task required that the model learn generalizations of lexical patterns, including polarizing language, parts of speech patterns, and grammar dependency patterns. This research further cements the importance of using linguistic patterns such as the point of view of the author when building models of identify deception as well as language. In addition, while we are not seeking to identify 'attitude' alone, it is likely that attitude is correlated with news veracity.

A paper that was written using the LIAR corpus is "Where is your Evidence: Improving Fact Checking by Justification Modeling" (Alhindi et al., 2018). This paper was researching the improvement to fact-checking that could be made by adding justification for the annotation. The authors used logistic regression and Support Vector Machines (SVM). They used Bag of Words unigram features and GloVe word embeddings to build these models. To complement these feature-based machine learning mechanisms, they used a bidirectional LSTM.

## 3 Dataset

We decided to use the Politifact LIAR dataset as the text corpus for our project. This dataset contains 12,836 claims that were made over Facebook, Twitter, or other social media sites. The dataset contains statements with political relevance. As a result, topics discussed include the economy, government decisions, elections, and health care. The statements in question were not just tweets, but spanned a wide array of media and therefore included statements of a variety of mediums. These included speeches, TV interviews, debates, social media postings. The claims are then annotated for accuracy, with 6 possible designations: "pants on fire", "false", "mostly false", "half-true", "mostly true", and "true".

In addition to the annotation, the data set has 14 variables with other relevant information to the statement produced. The variables include: subject of the statement, the speaker, the job title of the speaker, the state (assuming the speaker was American), the party affiliation of the speaker, the total credit history count (including the number of false and true statements the speaker is recorded to have made in the past), the context, and the extracted justification. This dataset was created for the aforementioned paper by Tariq Alhindi et al., "Where is Your Evidence: Improving Fact-checking by Justification Modeling" (Alhindi et al., 2018). Despite the availability of a variety of variables, we elected to look solely at the statement and its' label so as to

isolate the linguistic traits that can be used to predict truthfulness in a statement.

We decided to run all of the models not just using the given 6-label classification in the LIAR dataset, but also on a 2-label classification that we designed. This classification combined "true" "mostly true" and "half true" into a "true" category, and it combined the rest into a "false" category. As is shown in the next sections, this change gave us a much higher accuracy. We also think it makes more sense, because it is not entirely clear if there is a meaningful distinction between labels such as "half true" and "mostly true," or "false" and "pants on fire" (but since the data has been used in many studies with these classifications, we decided to keep them, and compare them to our modified labels).

# 4 Methods

## 4.1 Type of models

In much of the literature written on deception detection, choice of vocabulary is consistently identified as a key attribute in classifying fake news. In order to test this hypothesis and evaluate the extent to which classification depends on vocabulary features, we built SVM, SVC, Naïve Bayes, and k-NN models that use Bag of Words (BoW) and Bag of N-Grams (BoNG) to label social media news as true or fake. We elected to use these to compare the results of our LSTM and BERT (and the BERT derivatives RoBERTa, ALBERT, and DistilBERT) models. The blackbox nature of neural network architecture makes delving into the reasons for their decision making convoluted. These more traditional machine learning models do not have any neural network components and are therefore more interpretable.

### 4.1.1 Machine learning classifiers

The Naïve Bayes classifier is a function that, using conditional probabilites, assigns a class label $\hat{y} = C_k$ for some $k$ as follows:

$$\hat{y} = \underset{k \in \{1,\dots,K\}}{\operatorname{argmax}} \; p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)$$

The k nearest neighbors classifier is an algorithm that, given a data point $x_i$, receives the classification that is the mode among its $k$ nearest neighbors, where the $k$ nearest neighbors are calculated using the standard measure of Euclidean distance: $d(q,p) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$

where $p$ and $q$ are Euclidean n-dimensional vectors – in our case, n is the length of the total vocabulary across all tweets, and $p$ and $q$ are two rows of the matrix of token counts that represent two tweets.

The final classifier we used is a support-vector machine SVM. The SVM attempts to minimize the following expression:

$$\frac{1}{n}\sum_{i=1}^{n} \max\left(0, 1 - y_i(w \cdot x_i - b)\right) \quad + \lambda \|w\|^2$$

where we minimize with respect to $b$ and $\vec{w}$, where $\vec{w}$ is the normal vector perpendicular to the hyperplane containing $\vec{x}$ and $b$. The minimizers $b$ and $\vec{w}$ determine the classifier $\vec{x} \mapsto \operatorname{sgn}(\vec{w} \cdot \vec{x} - b)$

We also run an SVC, which is an SVM whose hyperplane margins have been edited by a parameter $C$, which we set to 1 after some trial and error. The purpose of setting a larger $C$ is to choose a smaller-margin hyperplane that better classifies the training points, and thus reduces misclassification.

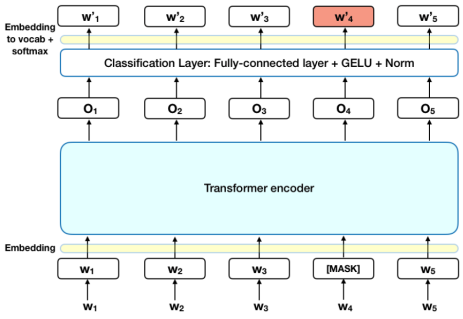
### 4.1.2 LSTM

We ran an LSTM using the Keras library in Python. We decided to create an LSTM because most of the literature about text classification, especially in the field of deception detection, use an LSTM as one of their models, and it would be interesting to compare its results to those from traditional machine

learning architectures and BERT. We used the standard GloVe (Global Vectors for Word Representation) embeddings – these vectors are created by taking the logarithms of words' probabilities in a global co-occurrence matrix, and then training a log-bilinear model to learn word vectors whose dot product equals these logarithms(Pennington et al., 2014).

#### 4.1.3 BERT

We used pretrained BERT models as a baseline, and then trained and tested them using our own corpus. We created these models using the simpletransformers wrapper made by Hugging Face, who offers a collection of pretrained neural network architectures compatible with PyTorch and TensorFlow (Horev, 2018). Given the ease of running various different BERT models once we created the baseline code, we decided to include RoBERTa and ALBERT models. RoBERTa is a BERT model pretrained on much more data, and it optimizes various other features of the model like batch size. RoBERTa tends to outperform most other BERT models. On the other hand, ALBERT (and similarly distilBERT) is a more efficient BERT model that reduces parameters by 80 percent. ALBERT still performs comparatively to BERT, sometimes exhibiting diminished accuracy. DistilBERT, another pared down BERT, has a similar trade-off between efficiency and accuracy. Below, we include a diagram of the BERT architecture for a classification task.




### 4.2 Data and parameter initialization

For the machine learning models, the main decision we made in terms of data cleaning was the choice of the stop word list. Stop words are common words such as 'and' and 'the' which are typically excluded from bag-of-words representations, because they don't actually add meaning to a classification. Based on literature review, we learned that the default 'english' stop words list that the Python library sklearn uses has many flaws (ex. excluding seemingly meaningful words like "computer" and excluding contractions but not their expanded form). Instead, we decided to use a stop words list called Onix, widely used and accepted in the literature.

For our LSTM, we again excluded stop words and performed standard data cleaning techniques, like removing line breaks and capitalization. Our embedding dimension was 100. We used an Adam optimizer and categorical cross-entropy (binary cross-entropy for the 2-label case) as the loss function. We ran the LSTM for 10 epochs with a batch size of 128. We ran the BERT models on similarly cleaned tweets for 10 epochs with a training batch size of 32 and a evaluation batch size of 64.

## 5 Results

### 5.1 Machine learning classifiers

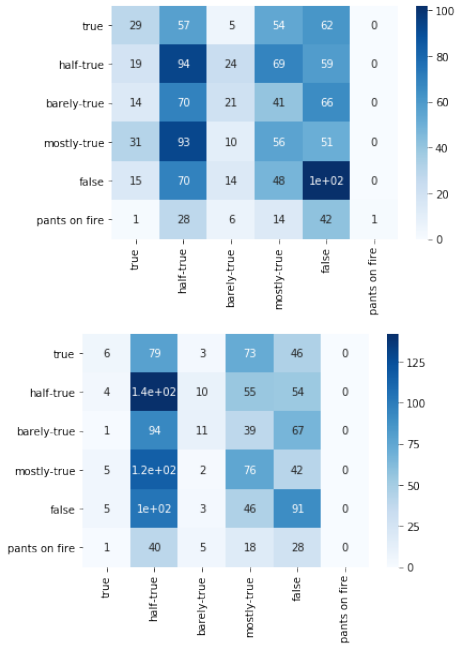
Table 1 displays the accuracies achieved when we ran various machine learning classification models. We used the Python sklearn package to convert a collection of text documents – tweets, in our case – to a matrix of token counts. We ran all four of these machine learning models with both unigram and then uni- and bi-gram bag of words representations. In the literature, SVMs are frequently the most accurate model for text classification. This proves to be the case here as well. The SVC

outperforms the SVM in most cases, as it is just a refined version of the same model. We also notice that the uni and bigram representation is more accurate than just the unigram version. This makes intuitive sense, as we are providing the model with more data – words and pairs of words – to learn on. One can imagine that pairs of words such "remarkably good" can speak to the credibility of a statement. Finally, we also confirm that the k-nearest neighbors model is the least accurate, as it is a more rudimentary machine learning task simply based on Euclidean distance.

Table 1: Machine learning accuracies

| | NB | k-NN | SVC | SVM |
|---|---|---|---|---|
| 6-class uni/bi-gram | 0.242 | 0.215 | 0.231 | 0.249 |
| 2-class uni/bi-gram | 0.607 | 0.584 | 0.628 | 0.593 |
| 6-class unigram | 0.235 | 0.228 | 0.239 | 0.221 |
| 2-class unigram | 0.604 | 0.579 | 0.619 | 0.570 |

We also decided to also create two heat maps of the confusion matrices for the 6-class SVC and SVM (following in that order):




It is interesting to note how different these two heatmaps are. The SVC seems to be misclassifying less frequently than the SVM (which makes sense, considering that is the purpose of the parameter $C$), but the SVM deals with the "half-true" classification better. This could just be by chance, or maybe because the greater margins for error of the SVM over the SVC allows it to capture fuzzy classifications like "half true," which is probably very similar to the "true" and "mostly true" cases.

### 5.2 Neural network models

We expected the BERT classifier to have higher accuracy than the machine learning classifiers. By its bidirectional transformer nature, and the various attention mechanisms it uses, BERT takes into account contextual information when making a classification decision, not just the vocabulary, which is what the bag-of-words machine learning classifiers use. Thus, if context provides meaningful information about the credibility of a tweet, BERT should outperform the machine learning classifiers. Table 2 displays the accuracies we achieved with our various BERT models.

Table 2: BERT accuracies

| | BERT | ALBERT | roBERTa |
|---|---|---|---|
| 6-class | 0.247 | 0.256 | 0.263 |
| 2-class | 0.639 | 0.642 | 0.649 |

These results are relatively surprising, especially in the 2-class case. We would expect BERT to significantly outperform the SVM. Indeed, many studies in the literature have BERT performing at over 90 percent accuracy for 2-label classification tasks. There could be many causes for our low accuracy. Some of these include the fact that we did not exclude stop words or maybe we did not achieve the best hyper-parameters. However, even with these modifications, the accuracy would not improve dramatically. This suggests to us that BERT might not be learning a meaningful syntactic and linguistic representation of the clas-

sification task, based on features like context and attention. The accuracies we achieved through our LSTM were almost the same as the BERT and machine learning classifiers. Given that our LSTM was unidirectional and it achieved the same accuracy as BERT provides more evidence that BERT is not using its bidirectional structure to learn a contextual, linguistic representation of the classification task.

### 5.3 Prediction

#### 5.3.1 Sentence Prediction using BERT

In order to better assess the predictions being made by BERT, we looked at the classification values assigned to phrases that differed only by one linguistic feature. One such example, includes the juxtaposition between the tweets "Mexico is experiencing very big CoronaVirus problems.", which was labeled as true, and "Mexico is experiencing very big CoronaVirus problems. Classic.", which was labeled as false. In this example, we can see that the addition of the sentence "Classic" tips the prediction into falsehood. This is interesting, as varied sentence structure has been referenced as a feature of credible writing by relevant literature. Another example includes the input sentences, "Omar is a disaster who wants higher taxes.", which was labeled as true, and "Omar is a disaster who wants much higher taxes", which was labeled as false. In this case, incorporating the modifier, "much" changed the prediction. This is aligned with research that has been previously done hypothesizing that exaggeration and more dramatic statements imply deceit in texts. We also found that phrases that used of superlatives and comparative adjectives tended to be marked as fake news when compared to phrases that instead used a standard adjective. Another linguistic signal that previous literature identified as a potential predictor of false news was strong negative statements and phrases that express doubt. One such example that supported this hypothesis was the phrase "NATO is opening up a terror division. I am not going to get credit for it.", which was classified as false, while the tweet "NATO is opening up a terror division, I am going to get credit for it." was classified as true.

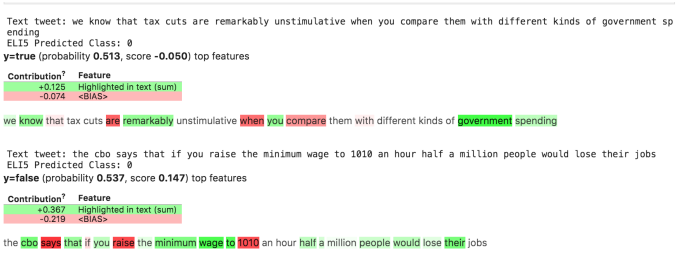
One concern that we had with the BERT predictions was the over-representation of tweets by President Trump in the dataset. As a result of Trump's social media popularity, there were many messages created by him that were labeled as false. One popular phrase Trump uses when referring to Joe Biden is, "sleepy Joe". We found that when fed the phrase: "Trump and Joe Biden are in the election", the model classified the phrase as true, whereas when fed the phrase, "Trump and sleepy Joe Biden are in the election", the model predicted the sentence to be a falsehood. The phrase, "sleepy Joe" is frequently used by Trump and his political supporters, but few Democrats would use the defamatory term, "sleepy Joe". "Sleepy" in itself is an objective adjective that carries a slightly negative attitude. The sentence, "The sleepy boy is in New York." was also marked as fake, despite any discussion of elections or Joe Biden. This suggests that the model learned the term 'sleepy' to be loaded language signaling fake news. This suggests that the model learned to predict tweets that contain phrases frequently used by Trump to be false. This would create a bias against Republican political figures that may be using phrases or messages that are akin to those in Trump's tweets.

#### 5.3.2 Understanding distilBERT with LIME

LIME is a model agnostic technique that contributes to model interpretation (Ribeiro et al., 2016). The goal of LIME is to provide insight into network predictions so as to im-

prove transparency in machine learning predictions. We applied LIME to our distilBERT classifier techniques in order to get a sense of the key features used in making predictions. LIME systematically generates small perturbations in the training dataset and observes how the changes influence the prediction process. This system allows LIME to identify the feature vectors that were most influential in the assignment of a given prediction label. LIME then proceeds to output the the set of features that had the greatest impact on predictions. This method works well with bag of words and bag of n-grams, as they are not dense techniques, and can be applied to traditional machine learning classifiers as well as neural networks. LIME has been criticized for using sparse linear local approximations, which may not be appropriate for models that are highly non-linear. We used the LIME algorithm to generate the following visualizations of our distilBERT classifier.

| Contribution? | Feature |
|---|---|
| +0.806 | we know |
| +0.763 | with different |
| +0.588 | remarkably |
| +0.318 | are |
| +0.301 | that tax |
| +0.286 | different kinds |
| +0.257 | you |
| +0.250 | cuts |
| +0.225 | unstimulative |
| +0.225 | compare |
| +0.139 | when |
| +0.100 | know that |
| +0.094 | compare them |
| +0.066 | you compare |
| +0.047 | of government |
| +0.001 | <BIAS> |
| -0.036 | know |
| -0.075 | of |
| -0.085 | we |
| -0.088 | cuts are |
| -0.106 | spending |
| -0.148 | tax |
| -0.158 | unstimulative when |
| -0.164 | remarkably unstimulative |
| -0.222 | tax cuts |
| -0.233 | kinds |
| -0.237 | different |
| -0.255 | them with |
| -0.272 | that |
| -0.292 | are remarkably |
| -0.394 | with |
| -0.657 | when you |




Here, LIME takes a bag of unigrams and bag of bigrams approach to determining which words in the tweet were most significant to contributing to the ”true” classification. One promising observation is that ”when you” is classified as significantly detracting from the ”true” label. In the literature, it is mentioned that the use of second-person pronouns mark a sentence as potentially less credible, and more emotional and subjective. On the other hand, the first-person ”we” is marked as a significant contributor to the ”true” label. The word ”government” is marked as a significant true contributor, which makes intuitive sense, and could suggest that the model is learning through vocabulary. Although these are linguistically meaningful results, this table has some clear inconsistencies, such as labeling ”remarkably” as a true contributor, but ”are remarkably” as a true detractor. Some other words marked as important do not have any real significance – like ”says.” This could be because we did not train LIME on enough samples – the baseline is 5000 samples, which takes a significant amount of CPU and RAM. We ran it only on 100 samples, probably resulting in worse predictions and explanations. Another way to improve LIME’s analysis and BERT’s performance would be to effectively and meaningfully remove stop-words. This would at least avoid the ”remarkably” and ”are remarkably” inconsistency. This is a future measure we will take to improve the study.

## 6 Conclusions

This study offers novel analyses of the LIAR dataset using a confluence of state-of-the-art neural network models and machine learning classifiers. Based on our similar accuracies between the bag-of-words classifiers, like SVM, and BERT, we suspect that vocabulary is a signifiant contributor to BERT's classification process. On the other hand, we do not see as much evidence that BERT is making use of context and attention to make more accurate credibility classifications. To further explore this question, a future step would be to visualize the attention mechanism that BERT is using. We would also improve our BERT by excluding stop words and further tuning hyper parameters. We also hope to hand encode other linguistic features, like polarity, second person pronoun frequency, and grammatical structure (like distance between certain words and a second person pronoun) into machine learning classifiers, and see if we obtain higher accuracy than the bag-of-words representation.

## References

Tariq Alhindi, Savvas Petridis, and Smaranda Muresan. 2018. Where is your evidence: Improving fact-checking by justification modeling. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 85–90, Brussels, Belgium. Association for Computational Linguistics.

Giovanni Da San Martino, Seunghak Yu, Alberto Barrón-Cedeño, Rostislav Petrov, and Preslav Nakov. 2019. Fine-grained analysis of propaganda in news article. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5636–5646, Hong Kong, China. Association for Computational Linguistics.

Ahmed Hassan, Vahed Qazvinian, and Dragomir Radev. 2010. What's with the attitude? identifying sentences with attitude in online discussions. pages 1245–1255.

Rani Horev. 2018. Towards data science.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should i trust you?": Explaining the predictions of any classifier.