An Adversarial Attack on Fake News Detectors

Joshua Bundt
Department of Computer Science
Northeastern University
Boston, MA
NUID: 001426739
bundt.j@husky.neu.edu

Christopher Stadler
Department of Computer Science
Northeastern University
Boston, MA
NUID: 001891582
stadler.c@husky.neu.edu

1 Introduction

In the wake of the 2016 U.S. Presidential Election and Russian interference in the United States political process via information operations, social media companies and government organizations are pursuing technical solutions to prevent or limit the spread of fake news. Research since the 2016 election has revealed that these information operations were highly effective. An in-depth look at the propagation of fake news done by Shao et al. [1] concludes that six percent of Twitter accounts identified as bots were primarily responsible for spreading 31 percent of "low-credibility" information on the social network. In response to the accusations of fake news being promulgated on Facebook, the company allegedly now employs 20,000 "News Feed Data Integrity Specialists" in order to fight the misuse of the social media site. A first step in stemming the creation and spread of fake news is accurately identifying it in the wild.

The Fake News Challenge (FNC) was launched in 2017 by a group of researchers from academia and industry in response to this crisis with the following goal [2]:

> To explore how artificial intelligence technologies, particularly machine learning and natural language processing, might be leveraged to combat the fake news problem. We believe that these AI technologies hold promise for significantly automating parts of the procedure human fact checkers use today to determine if a story is real or a hoax.

If machine learning models are going to be used to combat fake news then they must perform well not only against the current types of fake news articles, which are designed to fool human readers, but also against articles which are specifically crafted to fool the machine learning models themselves — adversarial examples.

We attempt here to test the strength of machine learning based fake news detectors by measuring their performance against adversarial examples.

1.1 Challenges

When constructing an adversarial example there is always a trade-off between fooling the classifier and preserving the semantic content perceived by humans. In the context of fake news the author's goal is to convey some (misleading) information to readers. Any alterations to the article that are made to fool a fake news detector must not disrupt the content, or else the original purpose of the article will not be achieved.

Prior work has been very successful at constructing adversarial images which are still recognizable to humans. These methods work by modifying many pixels in very small increments so that the changes are imperceptible to humans [3]. This approach is possible for images because humans cannot observe small changes to pixel values. Even changing the red component value of every pixel in a pixel is not easily seen by human eyes. On the contrary, in the text classification domain it is not possible to make such small alterations to the input without being noticed because humans can easily recognized with words are changed. Additionally, only certain combinations of words in a specific order will make sense to the reader. In images, there are numerous combinations and even orders to pixel values that will all be recognized as the same image. Therefore we cannot simply calculate the gradient and use that directly to alter the input.

1.2 Our contributions

- We develop a framework for constructing text adversarial examples using a *generative model*, but which does not depend on any internal knowledge of this model.
- We test examples generated by the generative model against black-box *target models*.

2 Related Work

The accuracy of a machine learning model depends on the assumption that it's training data is drawn identically and independently from the same distribution as the examples it will encounter during operation. However, when the model is used in real world applications (a economical and geopolitical competitive environment) then the distribution of data is not fixed and likely manipulated by an adversary in order to purposefully produce errors or degrade the performance of the model.

Adversarial examples became prominent when Szegedy et al.[3] found that neural networks are vulnerable due to "blind spots" and created adversarial examples that exploited these blind spots by maximizing the prediction error with imperceptible non-random perturbations to images. Goodfellow et al [4] expanded this weakness in image classification models to show a generalization between trained models based on the property of linearization in the adversarial samples. They demonstrated generating adversarial samples for the MNIST dataset that fooled multiple models, even getting various models to agree on on the same mis-classified class.

Research by Goodfellow et al.[4] demonstrates that humans can easily correctly classify an adversarial image sample which makes the results startling. Naively it appears that highly complex DNNs are easily fooled by images of a panda which a three year old could correctly identify. In contrast to the ease of classification in imagery, stance detection in text is not such an simple task for humans. Research done by Hanselowski et al [5] following the Fake News Challenge found that their human subjects, were only able to correctly identify stance labels (agree,disagree, discuss, unrelated) on the dataset with an F1 score of 75%.

The first work applying adversarial examples to text classification was published by Liang et al. in April 2017 [6]. They demonstrated that DNN-based text classifiers are prone to adversarial attack using a three-pronged strategy that included insertion of phrases, deletion of words, and replacement of characters in a document. They found that all three strategies were required to be used in combination to achieve a successful misclassification. In their research using the DBpedia ontology dataset, human subjects were able to correctly classify 10 original samples with an average accuracy of 94.2% before perturbation, and could only distinguish adversarial samples from originals with an accuracy of 5%. Unlike this work we use only the strategy of replacing words, and do so by using synonyms instead of replacing individual characters to create misspellings.

Research by Samata and Mehta [7] refined the approach of Liang et al. and contributed the idea of altering the most important words in the source text, as measured by their contribution to the class probability. They applied their method of synthesizing adversarial text samples again to the Twitter User Gender Classification dataset and also to the IMDB movie review dataset. We adopt the approach of targeting the words which contribute the most to the class probability from their work, but use a different algorithm for modifying these words, and test the generalization of examples produced by our method.

Goodfellow et al. first examined how adversarial examples can generalize to multiple models [4]. They generated adversarial images using one model trained on the MNIST dataset, and then demonstrated that other models trained on the same dataset were also fooled by these examples. Our approach is similar to this, but applied to the text domain.

Table 1: Replacing words with synonyms without altering meaning

Stance	agree
Headline	“Crude Sees a Bid on Reports of Saudi Arabia Pipeline Explosion”
Original body	"... And sparked a $2 surge in WTI crude. ..." "... A pipeline explosion will automatically decrease production. ..."
Altered body	"... And sparked a $2 surge in WTI petroleum. ..." "... A pipeline detonation will automatically decrease production. ..."

Table 2: Replacing words with synonyms but changing meaning

Stance	discuss
Headline	“Here’s What You Can Do On Path, The Social Network That Apple May Be About To Acquire”
Original body	“... Apple is on the verge of announcing an acquisition of social network and corresponding app Path. ...”
Altered body	“... Apple is on the verge of announcing an acquisition of societal web and corresponding app way. ...”

3 Methodology

Given a sample s with true label y and a classifier F we aim to generate a sample s' such that $F_y(s') < F_y(s)$ and $y = y'$, where F_y gives the class probability for y, and y' represents the true label of s'. Since our goal is to reduce the class probability $F_y(s)$, and not to change the label such that $F(s') \neq F(s)$, our examples are less specific to the model used to generate them. For example, even if $F(s) \neq y$ we can still use F to reduce the class probability, and potentially change the prediction of other models which correctly classify s.

To generate s' we first calculate the contribution of every word w_i in s to the class probability:

$$C_F(w_i, y) = F_y(s) - F_y(s^{w_i}) \tag{1}$$

Where s^{w_i} is the text produced by removing w_i from s.

In order of decreasing contribution we then replace each word w_i with a synonym. If no synonym can be found then w_i is not modified. This proceeds until K words have been replaced. K is a fixed parameter of the algorithm which determines the number of changes which will be made.

Because we directly compute the contribution of each word to the class probability our technique does not rely on any knowledge of the structure of the classifier. Any model which provides class probabilities can be used as the generative model in this architecture.

Synonym selection First, we tag each word in the document with its part of speech using the Natural Language Toolkit for Python [8]. Then, for any given word, we collect all synonyms for the word and its part of speech using the NLTK interface to the WordNet lexical database [9]. Because many of the synonyms are different forms of the original word (singular/plural, verb tense, alternate spellings, etc.) we remove synonyms which have an edit distance of 3 or fewer from the original word (calculated using NLTK).

Even though, by definition, synonyms should have the same or very close meanings, we hypothesize that it is more difficult for machine learning models than for humans to recognize the common meaning behind the concrete representations. This is similar to how adversarial images demonstrate the difficulty machine learning models have perceiving high-level structures which are clear to humans.

Table 1 hows an example where substituting synonyms does not substantially alter the meaning that is perceived by humans. In table 2 though we can see how our strategy sometimes does not meet this standard. Here, replacing each word with a synonym loses the context: "social network" has semantic content apart from that of each individual word, and "Path" is used here as the name of a company.

Table 3: Distribution of dataset

Total count	Agree	Disagree	Discuss
7064	26.9%	9.9%	63.2%

Table 4: Classification of original examples by the baseline model.

	Prediction			
Truth	agree	disagree	discuss	unrelated
agree	173	10	1435	285
disagree	39	7	413	238
discuss	221	7	3556	680

4 Experimental Results

Dataset For our starting examples we used the test dataset from the first Fake News Challenge competition (FNC-1). This was not released until after the end of the competition and so the models were not trained on this data. We excluded examples classified as *unrelated* because they are not relevant to our threat model: we assume an *unrelated* article would be ignored by a fake news detector. This leaves 7064 samples. Each sample consists of a headline, article body, and the stance label — one of *agree*, *disagree*, or *discuss*. The distribution of the classes is shown in table 3.

Generative model To generate adversarial examples we used the baseline model provided by the organizers of the FNC-1 competition [10]. This is a gradient boosting classifier composed of 200 decision trees. This was trained on the competition training dataset. The features it uses are the frequencies with which ngrams from the headline appear in the body, and frequencies of specific manually chosen words related to stance ("not", "fake", "deny", etc.). On our unaltered dataset this model achieved an accuracy of 60.4% (see table 4).

Target model To test how our generated adversarial examples generalize we tested them against another model from the FNC-1 competition. Several of the models are open source and of those we chose the UCL Machine Reading team's submission, which placed third overall. This choice was mostly technical — we were unable to run the top two submissions. The UCL model consists of a multi-layer perceptron with a single hidden layer of 100 units [11]. Its features consist of term frequency and term frequency-inverse document frequency vectors for both the headline and body, as well as the cosine similarity of the two TF-IDF vectors. On our unaltered dataset this model achieved an accuracy of 70.99% (see table 5).

Baseline model results Table 4 shows the predictions of the baseline model against our transformed dataset, with $K = 4$. Overall, the accuracy decreased to 51.03%. We can see that we were most effective at causing *agree* and *disagree* samples to become misclassified.

UCL model results Table 7 shows the predictions of the UCL MR model against our transformed dataset, with $K = 4$. The overall accuracy decreased to 67.03%. While this is small, we were more effective specifically at causing *disagree* samples to be misclassified, reducing the accuracy in that class by 45%.

Table 5: Classification of original examples by UCL model.

	Prediction			
Truth	agree	disagree	discuss	unrelated
agree	793	110	939	130
disagree	158	49	327	135
discuss	500	43	3633	383

Table 6: Classification of adversarial examples (K=4) by the baseline model.

	Prediction			
Truth	agree	disagree	discuss	unrelated
agree	82 (-52%)	12	1317	461
disagree	23	1 (-86%)	338	307
discuss	215	12	3010 (-15%)	1222

Table 7: Classification of adversarial examples (K=4) examples by UCL model.

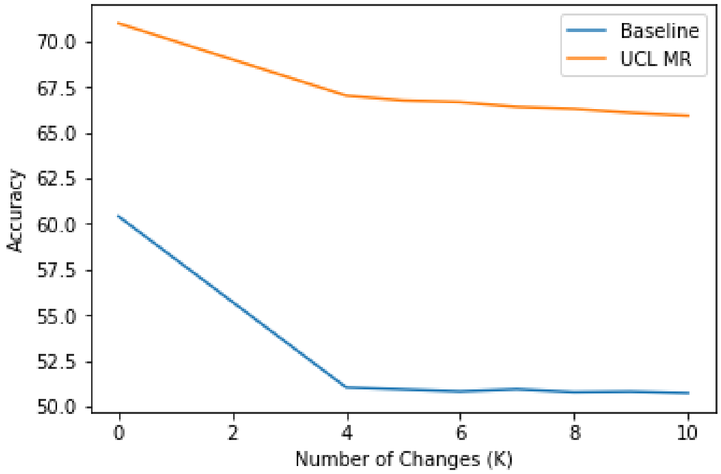
	Prediction			
Truth	agree	disagree	discuss	unrelated
agree	737 (-7%)	8	917	210
disagree	147	27 (-45%)	315	180
discuss	474	35	3454 (-5%)	496

Changing K Figure 4 shows that increasing K does not significantly decrease the accuracy of the models. We hypothesize that this is due to the models' dependence on a small number of keywords, such that after replacing the most contributing keyword, replacing additional keywords does not alter the label. It is worth noting that while the baseline model's accuracy plateaus at $K = 4$, there is a slight decrease in the accuracy of the UCL MR model. This shows that we are able to continue extracting useful information from the generative model even after it's prediction is incorrect.

Discussion We have demonstrated a technique for generating adversarial examples for text classification problems, and shown that these examples can generalize across models. Although the generated examples are not as effective against the target model, this could likely be improved by incorporating additional techniques for transforming the examples, such as inserting text, as was done [6].

The heavy dominance of the *discuss* label in the dataset is unfortunate for our use case of creating adversarial samples. Demonstrated in our results above, we have more success in changing a *agree* or *disagree* label than we do changing with *discuss* labels. In an adversarial scenario, it might be most advantageous to change *agree* to *disagree* or vice-versa, but the dataset was limited in the number of such samples.

Figure 1: Generative and Target model vs K



References

[1] Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Kaicheng Yang, Alessandro Flammini, and Filippo Menczer. The spread of low-credibility content by social bots, jul 2017. ISSN 2041-1723. URL `http://arxiv.org/abs/1707.07592`.

[2] Fake News Challenge, October 2018. URL `https://web.archive.org/web/20181006141229/http://www.fakenewschallenge.org/`.

[3] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, dec 2013. URL `http://arxiv.org/abs/1312.6199`.

[4] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. *arXiv:1412.6572 [cs, stat]*, December 2014. URL `http://arxiv.org/abs/1412.6572`. arXiv: 1412.6572.

[5] Andreas Hanselowski, Avinesh PVS, Benjamin Schiller, Felix Caspelherr, Debanjan Chaudhuri, Christian M. Meyer, and Iryna Gurevych. A Retrospective Analysis of the Fake News Challenge Stance Detection Task, jun 2018. ISSN 0264-0414. URL `http://arxiv.org/abs/1806.05180`.

[6] Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. Deep Text Classification Can be Fooled. *arXiv:1704.08006 [cs]*, April 2017. URL `http://arxiv.org/abs/1704.08006`. arXiv: 1704.08006.

[7] Suranjana Samanta and Sameep Mehta. Towards Crafting Text Adversarial Samples. *arXiv:1707.02812 [cs]*, July 2017. URL `http://arxiv.org/abs/1707.02812`. arXiv: 1707.02812.

[8] Natural Language Toolkit — NLTK 3.4 documentation, 2018. URL `https://www.nltk.org/`.

[9] Princeton. WordNet:A Lexical Database for English, 2018. URL `https://wordnet.princeton.edu/`.

[10] A baseline implementation for FNC-1, November 2018. URL `https://github.com/FakeNewsChallenge/fnc-1-baseline`.

[11] Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis, and Sebastian Riedel. A simple but tough-to-beat baseline for the fake news challenge stance detection task. *CoRR*, abs/1707.03264, 2017. URL `http://arxiv.org/abs/1707.03264`.

A Source code

All source code for this project can be found at `https://github.com/CJStadler/fnc-adversarial`.