Black-Box Generation of Adversarial Text Sequences to Evade Deep Learning Classifiers

1. **What is the primary lesson(s) you took away from this paper (avoid abstract level summary)?**

   The authors in this paper presented an algorithm called DeepWordBug that effectively generates small text perturbations in a black-box setting (the attackers are unaware of the model architecture, parameters, or training datasets) that forces a deep-learning classifier to misclassify a text input by applying simple character-level transformations to the highest ranked words to minimize the edit distance of the perturbation. The authors also developed a few scoring strategies/methods to find the most important words in the input space to modify such that the deep classifier makes a wrong prediction. These scoring strategies are Temporal Score (TS), Temporal Tail Score (TTS), and Combined Score (TS+TTS).

   Additionally, the authors evaluated their algorithm, DeepWordBug, on two real-world free text datasets: Enron Spam Emails and IMDB Movie Reviews. They trained an RNN model to classify movie reviews into positive and negative classes. At the same time, they trained another RNN model (LSTM) to build a spam filter that can determine whether a particular message is spam or not. Without adversarial examples, the first model achieves 84% accuracy on the IMDB dataset and 99% on the Enron Spam dataset. When using the adversarial examples generated by the authors' algorithm, the first model achieves 26% with the change of 20 words per movie review using the Combined Score (TS+TTS) and 44% accuracy on the Enron Spam dataset with the same settings.

   Lastly, the authors evaluated the transferability of the adversarial attacks (sequences) by feeding adversarial sequences generated by one RNN model to another RNN model on the same task. They found that most adversarial samples could successfully transfer to other models, even to those models with different word embeddings.

2. **Could I have done this work if I had the idea why or why not?**

   I do not think I could have done this work if I had the idea, given my limited knowledge of RNN models (I am taking CS570 Deep Learning class this semester). However, the high-level idea in this work could be implemented and demonstrated using any architecture, which (1) using a scoring function to determine the importance of every word to the classification result and rank the words based on their scores, and (2) using a transformation algorithm to change the selected words.

3. **Is there anything I could do to repeat or validate? (Make notes of materials available from the paper- data sets, source code).**

   Yes, the authors are open-sourcing their attack (DeepWordBug) and models, and they publicly published their attack and models on GitHub here: https://github.com/QData/deepWordBug.

   The authors also used two free datasets: Enron Spam Emails (https://www.cs.cmu.edu/~enron/) and IMDB Movie Reviews (https://ai.stanford.edu/%7Eamaas/data/sentiment/).

4. **What is my best idea for follow on work that I could personally do?**

   My best idea for the follow-on work that I could personally do is to first repeat the same work using their English datasets to develop a better understanding of how the attack works. Secondly, I could repeat the same work or part of it (sentiment analysis) using Arabic sentiment analysis datasets, examine the generated results, and test if this kind of adversarial text attack works in different languages like Arabic.

5. **What is my best idea for follow on work that I'd like to see the authors or others do?**

   I do not have an idea for the follow-on work I would like to see the authors or others do.

6. **Any logistical experimental lessons I learned (this is a little different than #1 here you are looking for techniques)?**

   I liked how the authors evaluated the transferability of the adversarial attacks (sequences) by feeding adversarial sequences generated by one RNN model to another RNN model on the same task. They found that most adversarial samples can successfully transfer to other models, even to those models with different word embeddings.

7. **How does this compare to the other papers we read? Most similar? How different? Other comparisons?**

   My first paper to read was Alzantot et al. (2018). This paper, Gao et al. (2018), is similar to it in terms of the type of the attacks, where both are black-box attacks (the attackers are unaware of the model architecture, parameters, or training datasets). Both papers are different in terms of space and method. Alzantot et al. (2018) was in the embedding space, whereas this paper was in the input space. Alzantot et al. (2018) used the word-level replacements method, whereas this paper used the character-level replacements method

8. **What is your biggest criticism of the paper?**

   One big criticism of the paper is that the authors explained their three different score functions: temporal score, temporal tail score, and combined score, in this paper. They added a fourth method called "Replace-1" that they did not explain in the whole paper (only mentioned twice in: the list of used methods and the results of the used methods). I had to look up this method and found that it is one of theirs, but they have only explained it in their arXiv.org long version – extended report: (https://arxiv.org/pdf/1801.04354.pdf).

   Another minor criticism is that this paper has five tables and seven graphs, all on six pages. I felt the paper needed more organizing, and a couple of tables/graphs could have been dropped or merged.

9. **List 3 cited references or terms/concepts that you would be most interested in reading/learning more about.**

   I am interested in reading the authors long version or extended report on the arXiv.org (https://arxiv.org/pdf/1801.04354.pdf) to understand their attack and methods deeply especially the "Replace-1" method that has not been explained in this paper.

   I am also interested in reading these two references to understand the evaluation of the authors' methods, where they compared their methods results with these two references/papers' results:

   N. Papernot, P. McDaniel, A. Swami, and R. Harang, ***Crafting adversarial input sequences for recurrent neural networks.*** *In Military Communications Conference, MILCOM 2016-2016 IEEE. IEEE, 2016, pp. 49–54.*

   S. Samanta and S. Mehta. ***Towards crafting text adversarial samples.*** *arXiv preprint arXiv:1707.02812, 2017.*

   I would like to learn about Euclidean distance, which calculates the distance between two vectors/data points. This distance has been used heavily in these adversarial text attacks to measure the distance between words or embeddings.