Detecting Machine Generated Text using Neural Networks

Luan Fletcher, supervised by Prof Jason Wyse

July 26, 2021

Abstract

In this project we aim to build a classifier capable of distinguishing between machine-generated text and human text. We also aim to explore how factors (for example text length) affect classifier performance.

1 Introduction

Over the past decade, language models capable of generating "human-sounding" text have been developed. One such language model is known as Generative Pre-Trained Transformer 2 (GPT-2). This model is freely available online and is relatively easy to use. The creators of this model, OpenAI, have expressed concerns that it could be used by bad actors for the purposes of, for example, flooding social media with convincing machine generated posts [1]. In this project, we demonstrate that it is possible to distinguish between GPT text and human text with fairly good accuracy. We also explore in which contexts a classifier of this kind may struggle.

2 Background

We will first briefly explain some concepts discussed in the rest of the report.

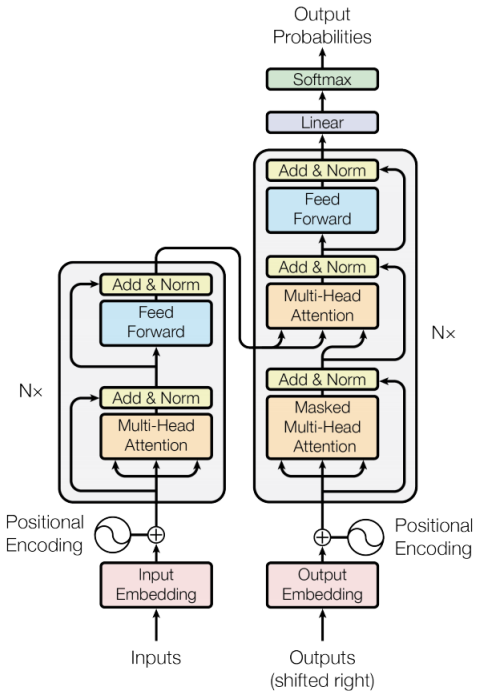
2.1 Word Embeddings

Word embeddings are representations of words as a vector of numbers. These vectors in some sense encode the "meaning" of the word, such that words which are closer in meaning are also closer in the embedded vector space. For example, the words "dog" and "puppy" would be closer in this space than the words "dog" and "England". Embeddings are frequently used in natural language processing as input for statistical models/neural networks etc. Embeddings can have some fairly interesting and surprising properties. Vector addition and subtraction can have meaningful results which "make sense". In the pre-trained embeddings used in this project (GloVe [2]), for example, the vector for "king" minus the vector for "man" is closest to the vector for "queen".

These embeddings can be obtained in a number of different ways. One method (known as Continuous Bag of Words) is to assign words similar vectors if they are found in similar contexts in some large corpus of text. For example, the words "dog" and "cat" will likely be seen often near the words "petted" or "vet" etc., and CBOW will therefore assign the words "dog" and "cat" similar vectors.

2.2 Transformers

A Transformer [3] is a neural architecture used in the processing of sequential data such as text. The following diagram from the original paper introducing Transformers is a good way of understanding their mechanism.

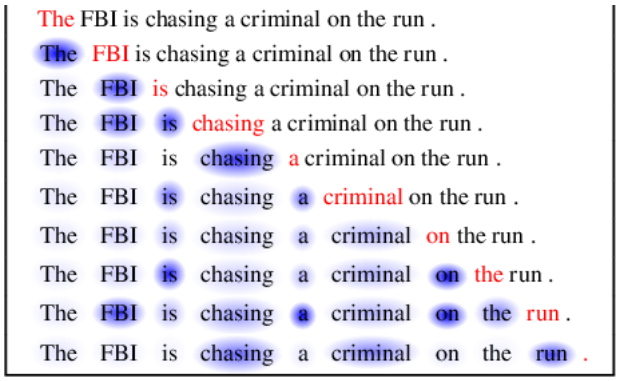


The left hand side consists of the "encoder" and the right hand side consists of the "decoder". Transformers were originally designed for sequence2sequence tasks such as translation, so the encoding block would, for example, create some embedding of a sentence to be translated and the decoding block would convert that embedding into a sentence in the target language.

For our task, we use BERT, which is a modified version of the original Transformer architecture that essentially discards the decoder and allows us to use the

raw output of the encoder. In this project, we use this raw output as the input for a simple feedforward neural network which we train on the task of classifying text as machine generated or human. This raw output can be thought of as an embedding of the entire piece of text, taking into account context, structure, syntax etc.

The most important part of the transformer architecture are the multi-head attention blocks. These are used to allow the model to "focus" on certain parts of the text which are related to one another. "Multi-head" simply means that attention is applied multiple times on the input text to capture different dependencies. An example visualisation is given below. Here, as the text is processed word-by-word, we can see that the model is "focusing" on different parts of the context. For example, when processing the word "chasing", the model essentially discards the word "The" as unimportant and instead gives more weight to the words "FBI" and "is".



3 Methods

We began by obtaining a large amount of training data. We then proceeded to train two classifiers on the task of distinguishing between GPT-2 and human text. One was a simple baseline and one was a BERT model [4].

3.1 Training Data

We obtained 4 large corpora of text (news headlines [5], news articles [6], Facebook comments [7], Tweets [8]) and finetuned a GPT-2 model on each of the corpora. Finetuning means that GPT-2 outputs text that "sounds like" it comes from a particular corpus. For example, the GPT-2 model we finetuned on the corpus of tweets began to output text that was similar in form, content and style to tweets.

We then randomly sampled 18,000 from each corpus for a total of 72,000 human text examples. Using each finetuned GPT-2 model, we generated 18,000 machine generated examples from each corpus, for a total of 72,000 machine generated text examples. We varied the temperature of the GPT-2 models as we generated this training data. Temperature is roughly a measure of the "randomness" of the generated text. We did this to see how our classifier's accuracy would be affected by the temperature of the GPT-2 model that generated the piece of text it was classifying.

3.2 Baseline

Our baseline model consisted of a simple logistic regression fitted on "average" word embeddings. The word embeddings we used were a set of 300-dimensional embeddings (GloVe) pre-trained on a very large corpus of English text. Each training example (i.e. each tweet, news article etc.) was split into individual words and each of these words were converted into a GloVe vector. The logistic regression was then performed on the average word vector for each piece of text.

3.3 BERT

The main classifier we trained was a BERT model (as discussed in the background section). We used a Google Colab notebook for GPU acceleration. We also made use of the huggingface transformers Python library [9]. We trained the model for 3 epochs (OpenAI recommend 2-4 epochs of training) and selected the checkpoint with the highest accuracy on the test set for further evaluation.

4 Results

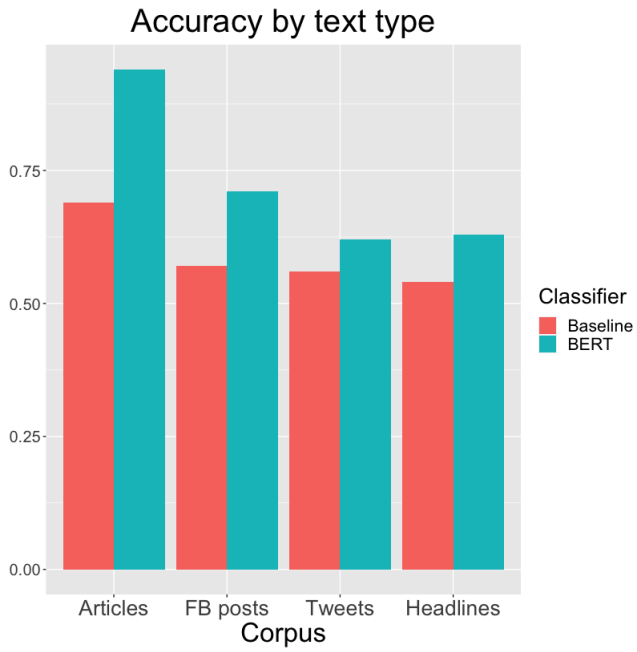
The headline accuracy numbers on the test set were as follows:

- **60%** for baseline
- **76%** for BERT

The headline accuracy numbers serve mainly to show that BERT gives a very significant improvement in performance for this task over baseline. It is likely that with more epochs of training/the use of a larger BERT model we could achieve higher accuracy on this task, so 76% is likely not the ceiling on accuracy for this particular set of data.

4.1 Accuracy by corpus

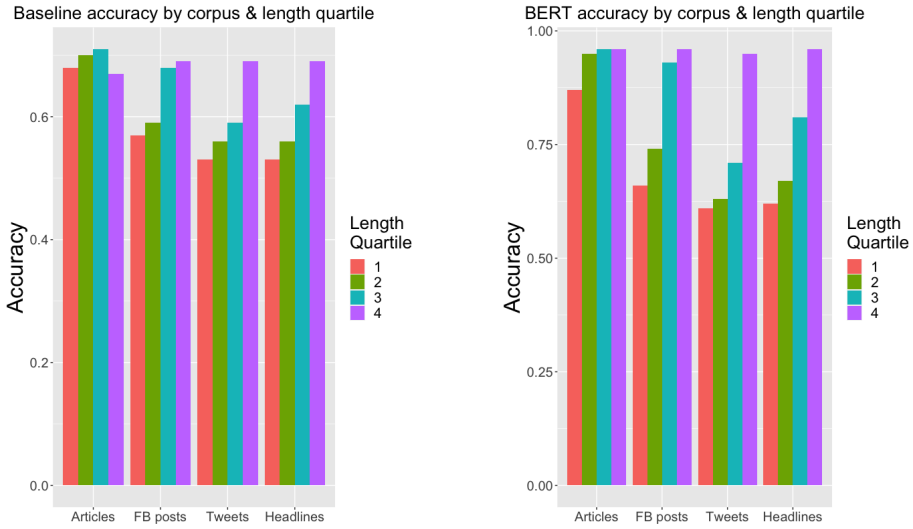
The following barplot shows accuracy of baseline and BERT by corpus.



We can see that for both baseline and BERT, accuracy varies significantly by corpus. Accuracy seems roughly to increase with the length of the text in each corpus. This hypothesis makes some sense intuitively, as if the text we are analysing is longer, there is a greater chance that GPT may "slip up". There is still a chance that, for example, news articles may be somehow inherently "easier" to classify and just coincidentally are longer. To rule this out, we check accuracy by length within corpora in the next subsection.

4.2 Accuracy by corpus and by length

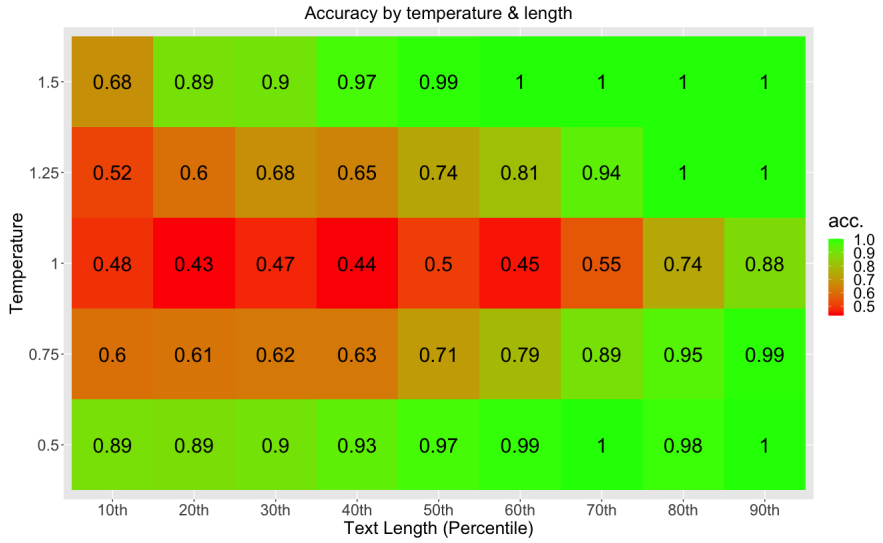
The following barplots show baseline and BERT accuracy by corpus and length quartile.



We can see here that even within corpora classification accuracy generally increases with the length of text. We can conclude from this that length is an important factor for classification accuracy regardless of which corpus the text comes from.

4.3 BERT Accuracy by length and temperature

The following heatmap shows BERT accuracy by length decile and temperature.



We can see clearly here that temperature and length have a very significant effect on BERT's accuracy on this task. Classification accuracy is very signifi-

cantly lower for text generated at a temperature of 1. It is also lower for shorter pieces of text. We can even see here that classification accuracy is only about chance for short pieces of text generated at a temperature of 1. We will discuss the implications of this further in our conclusion.

5 Conclusions

Our main conclusion is that it is possible to distinguish between GPT text and human text using an automatic classifier, but this comes with some significant caveats. A classifier of this kind will find it much more difficult to deal with short pieces of text. It may well be that there is some hard ceiling on the ability to detect machine generated text for very short pieces of text. There is simply very little information in, for example, a one or two sentence tweet. For this reason, it would appear that OpenAI's concerns about GPT mentioned in the introduction are likely well founded. This seems particularly true when it comes to the possibility of flooding a form of social media that consists of very short pieces of text (such as Twitter) with fake posts.

We also found that the parameters of the GPT model can affect the ability of the generated text to "fool" a classifier. We only tested temperature here, but it seems plausible that, for example, GPT could generate more convincing fine-tuned text with more training data on which to perform the finetuning etc.

The relative performances with respect to text length/temperature are more important here than the headline accuracy numbers. It is likely we could have achieved better accuracy with a larger BERT model or if we had trained the BERT model for longer, but this was not possible due to resource and time constraints.

6 Acknowledgements

I'd like to thank Prof Jason Wyse for his supervision and useful advice during this project. I'd also like to thank Prof Kirk Soodhalter for organising the internship and the Hamilton Trust for funding the internship.

References

[1] OpenAI, "Better language models and their implications," `https://openai.com/blog/better-language-models/`.

[2] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," `https://nlp.stanford.edu/projects/glove`.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, pp. 5998–6008, 2017.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[5] News Headline dataset (gathered by R. Kulkarni) `https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/0U9Z9F`.

[6] News Article dataset (gathered by A. Thompson) `https://www.kaggle.com/snapcrack/all-the-news`.

[7] Facebook Comment dataset (gathered by J. Bencina) `https://github.com/jbencina/facebook-news`.

[8] Tweet dataset (Sentiment 140 at Stanford) `http://help.sentiment140.com/for-students/`.

[9] huggingface transformers library `https://huggingface.co/transformers/`.