Distinguishing Texts Generated by ChatGPT from Human Generated

Samantha Tang
Department of Computer Science
University of Maryland, College Park
College Park, MD 20740
`stang10@umd.edu`

Christopher Tharratt
Department of Computer Science
University of Maryland, College Park
College Park, MD 20740
`tharratt@umd.edu`

Kyle Hassold
Department of Computer Science
University of Maryland, College Park
College Park, MD 20740
`khassold@umd.edu`

1 Introduction

ChatGPT is a supervised Reinforcement Learning from Human Feedback (RLHF) model and acts as a successor to the previous InstructGPT model [6]. The widespread availability and general effectiveness of ChatGPT has led the model to become a popular choice in various domains leading to a rise of issues in academic integrity and misinformation[8].

ChatGPT specifically uses RLHF to be a more effective learning model. RLHF incorporates human feedback as the model is trained, presenting generated responses to a human evaluator, then receiving feedback on how well the reponse aligns with their expectations. The feedback is then used to update the model's parameters, which in turn improves the model's ability to generate more human-like responses.

The inclusion of the supervised RLHF creates a model that can very effectively generate natural-sounding human text and can be extended to generate working programming solutions to given problems. This gives rise to a significant issue of academic integrity, as students working on assignments may use ChatGPT and other models to write the assignment for them. ChatGPT generates new text for a topic, that likely does not exist in full on the internet. This makes it difficult for instructors to detect when ChatGPT has been used on an assignment and violates academic integrity policies. Additionally, ChatGPT can be used to make believably sounding false information, or be used in online bot misinformation campaigns. Our team seeks to study various current and past strategies for detecting machine generated text and apply them to ChatGPT's model to detect instances where ChatGPT has generated a response rather than a human.

Our team has researched and evaluated several strategies for machine generated text detection, primarily used in the NLP space including frequency features, neural language model approaches, and human-aided methods. However, these methods have viewed the machine generated text detection problem as a binary classification problem; is the piece of text written by a human or not? Our work examines the problem through the lens of authorship verification.

Authorship verification is a category of natural language processing (NLP) problems that has existed prior to ChatGPT and similar RHLF models. The field of authorship verification consists of techniques and strategies designed to determine the original author of a piece of text, given examples of that author's writing. Authorship verification examines writing styles, flow, and context for the goal of determining and classifying any particular given text and differentiating it from the writings of other authors. There has been a lack of studies treating ChatGPT detection as an authorship verification problem, which will be the main theme of this project. The goal of our work was to create a model that could determine if a piece of text was written by a specific human author or ChatGPT by analyzing the writing styles of the specified author and ChatGPT itself.

2 Related Works

2.1 A Survey of Previous Detection Methods

As natural language generation (NLG) systems are becoming more accessible to all, questions regarding the security and potential abuse of these systems have arisen. These questions led to the development of the survey paper *Machine Generated Text: A Comprehensive Survey of Threat Models and Detection Methods* [2]. This paper discusses a variety of threats that NLG systems pose as a motivation for the need for methods of machine-generated text detection. The paper then surveys a wide array of current detection methods. For the purposes of our project, our analysis of this work will focus specifically on these detection methods presented by the survey paper.

The first category of detection approaches is referred to as *feature-based approaches*. Within this category are approaches that utilize statistical techniques and natural language processing to create feature vectors that are later classified using classification algorithms such as support vector machines, random forests, or neural networks. The focus of this category is the different ways to formulate the feature vectors.

The paper first identifies major types of *frequency features* which focus on the frequency of terms in a piece of text. Methods include frequency of tokens within text according to Zipf's law, term frequency - inverse data frequency unigram and bigram features, log-log lemma frequency versus rank, and n-gram overlap of words and parts-of-speech tags. Next, the paper discusses *fluency features* which focuses on the fluency and readability of text. Methods include the Gunning-Fog Index, the Flesch Index, proxy measures of coherence based on entities in important grammatical roles, and Yule's Q statistic for coherence. The paper then discusses *linguistic features from auxiliary models*. These features are more complex. Coreference resolution relationships, part-of-speech tags, and named entity tags are all utilized to measure the "consistency" of machine-generated text. Performing coreference resolution or assigning these tags requires processing with specialized auxiliary models that are often neural in nature. Next, the paper briefly touches on *complex phrasal features* stating that detection can be used to identify idiomatic phrases that were not commonly found in machine text. The last feature type discussed was *basic features*. Basic features are simple high-level characteristics of sentences such as the number of punctuation marks and length of sentences and paragraphs.

The second category of detection approaches is referred to as *neural language model approaches*. These approaches are based on neural networks and typically incorporate features derived from Transformer neural language models (NLMs). The paper separates this category into two types of approaches as they make up the majority of NLM-based machine-generated text detection.

The first NLM-based approach is the *zero-shot approach*. This approach uses generative models themselves, such as GPT-3 which is uni-directional, to detect either their own outputs or outputs from other generative models. These uni-directional generative models utilize token embeddings. Each token embedding is based on the embeddings of preceding tokens. As such, a feature vector can be created from the embedding of a classification token at the end of a sequence of tokens. With these feature vectors, labeled human and machine text can be used to train a layer of neurons in a neural network. The second NLM-based approach is the *fine-tuning approach*. In this approach, bi-directional language models are fine-tuned. The paper identified the RoBERTa model which was fine-tuned to differentiate between NLG model output and human-written NLG model training samples. For RoBERTa, training datasets were drawn from multiple different sampling methods to generalize more effectively to unknown sampling methods. Another method of fine-tuning uses attention map information from Transformer models to perform topological data analysis (TDA). The TDA is used as a feature for the detection of machine-generated text. The last fine-tuning method discussed is the use of energy-based models alongside a classifier of machine-generated text. Transformer architectures were utilized in various iterations of this method. They were initialized with pre-trained checkpoints and then fine-tuned on machine versus human classification datasets.

The last category of detection methods discussed by the paper was *human-aided methods*. While the previous two categories utilized purely automated methods, these approaches use a statistical or neural approach alongside a human analyst for review.

The first of these human-aided methods is the *Giant Language Model Test Room (GLTR)*. In this technique, the GLTR system augments human evaluators who are tasked with classifying a piece of text as human-written or machine-generated. The system augments these evaluators by highlighting tokens on the text that reflects the sampling probability of tokens for a Transformer model. Lastly, for the human-aided methods, the paper discusses *human performance in detection of language models*. In this section, the authors discuss various studies performed on humans tasked to classify texts as human-written or machine-generated to identify reasons for their classifications. These explanations may provide greater insight into identifying the boundaries between human and machine-generated texts for future detection methods.

In a summary of these categories of detection methods, the authors reiterate the key features of each method and, in some cases, discuss their generalized benefits and drawbacks. Feature-based detection methods have strength in providing diverse features which can complicate adversarial attacks and potentially lead to improvements in efficiency. The weaknesses of feature-based detection methods include the poor transferability of features across generation methods and sampling methods. Additionally, the paper notes these methods are most effective when longer collections of text are available. Neural based approaches are currently considered state of the art, with a trend toward bi-directional Transformer architectures. Potential improvements to these methods include incorporating other features beyond neural features to increase the difficulty of creating text adversaries to these methods. Lastly, human performance in the detection tasks is rather poor. However, there are insights into detection performed by humans that may assist future iterations of detection models.

2.2 Current Publicly Available Detection Tools

While there are many different approaches and tools developed to detect machine generated texts, there are a few common tools that are publicly available and widely used. These include GPTZero, OpenAI GPT-2 Output Detector, and AI Detector [1]. While these tools provide a user with a categorization of human or machine generated text and their confidence in their categorization, they are often susceptible to small changes in tested texts. For instance, additional paraphrasing was able to increase the confidence in the "realness" of the text by the OpenAI GPT-2 Output Detector by over 90 percent [1].

As such, while there are some widely available tools for the detection of machine generated text, they are often not reliable. As such, the previously surveyed detection methods need to be further explored for public use.

3 Methods

3.1 Dataset Selection

We initially identified several candidate datasets to train the Contriever model. One such candidate was a dataset from a University in Southwestern China (CEFLLUSC). It contains 400 compositions written by 100 English program students[4]. Each student was tasked with writing four compositions, each responding to the same four tasks. Initially, this seemed to be a very promising dataset to work with, as each of the authors responded to the same prompts, and we had access to the prompt information. However, analyzing the data raised some concerns due to higher than average inconsistencies in grammar, most likely due to the students learning English as a second language. Our concern was that any model trained on this dataset will pick up on grammar inconsistencies as the main indicator of ChatGPT vs Human responses, rather than content structure.

We decided instead to use the Reddit Cross-Topic Authorship Verification Corpus which consists of comments written between 2010 to 2016 from 1,000 Reddit users [3]. The data is taken from several different subreddits consisting of different topics, but similar to the previous dataset, contains four documents of text known to be written by the designated author (called "known documents"), and one unknown document that could be utilized for testing traditional authorship verification models. Three of the known documents were used for training our model, and the last known document was used for evaluation of our model.

We modified the Reddit Cross-Topic Authorship Verification Corpus to add ChatGPT as a Reddit "user" by generating four documents in a similar style to the known documents for each user in the corpus. The exact prompt used to create this ChatGPT user was “You are a Reddit user on the subreddit subreddit. You are responding to posts in news. Generate 20 different comments that could reasonably be found in subreddit by a Reddit user. Do not number responses. Do not include new lines. Responses should be unique and not generic. Make up posts that you are responding to.” This was done for four separate subreddits so that each document contained only the comments for the corresponding subreddit. This modification allows for contriever to be trained believing ChatGPT is a potential author for later detection.

3.2 Developing GPT Author Mimics

The aim of our model was to determine if a piece of text was written by a human author or written by GPT in the writing style of the human author. In order to evaluate this, we created a test bed of one text per each of the 1000 authors from the Reddit Cross-Topic Authorship Verification Corpus written in the specific author’s style. We refer to these texts as "GPT Mimics" as they are texts written by GPT in an attempt to mimic the specified author.

These GPT Mimics were created using the OpenAI Python API with the gpt-3.5-turbo model. The gpt-3.5-turbo model has a limit on the number of tokens that may be used in an interaction with the model. According to OpenAI’s help center, "[t]okens can be thought of as pieces of words". The token limit for the gpt-3.5-turbo model is 4097, which includes the number of tokens in the request to the model and its response [7]. On average the prompt we sent to the gpt-3.5-turbo model, which we will discuss in more detail later in the paper, was approximately 1600 tokens. To accommodate the token limit and the number of tokens in our request, the max_tokens parameter for the model’s response was set to 2300. Additionally, the model’s temperature parameter, which controls the randomness of responses from the model where 0 is the least amount of randomness and 1 is the most, was set to 0.5.

As stated above, for each user in the Reddit Cross-Topic Authorship Verification Corpus there were 4 documents labeled as known to indicate they were written by the specified author. We used three of these documents for training the model and set aside one for evaluation. We used these known document set for evaluation for each user to create the mimics. For each of the 1000 users, we gave the gpt-3.5-turbo model the following prompt:

> The following was written by Charlie. “<8000 bytes of the author’s known document set for evaluation>” Please mimic Charlie’s writing style and write on the same topic. You must write at least 800 words.

We chose an arbitrary author name, Charlie, for an easy reference to the author. In some cases, the model was unsuccessful in creating the mimic. These failures were either due to the model deeming the content of the text inappropriate or the model writing a response to the content of the author’s text rather than mimicking it. For these authors whose mimic was unsuccessfully created, we regenerated the mimic using one of four variations of prompts. Each variation was the same as the original prompt up through the inclusion of the excerpt of the author’s known document set for evaluation. Only the sentences following the excerpt of the original author’s text were varied. The regeneration process was continued with a different choice from the four variations until a successful mimic was created. The prompt variations are as follows:

> Prompt Variation 1: ...Do not respond to Charlie. However, please mimic Charlie’s writing style and write on the same topic. You must write at least 800 words.
> Prompt Variation 2: ...Can you write ramblings in the style of Charlie on the same topics. You must write at least 800 words.
> Prompt Variation 3: ...DO NOT respond to Charlie. Instead, write in the style of Charlie on the same topics using at least 800 words.
> Prompt Variation 4: ...Please mimic Charlie’s writing style and write on the same topic. DO NOT RESPOND TO CHARLIE. JUST ACT AS IF YOU ARE HIM. You must write at least 800 words.

The responses provided by GPT did not always contain the 800 words we requested, but they did contain enough tokens to be analyzed by our model.

As an empirical example of the quality of the GPT Mimics, we have included an example for the author "backpackwayne" from the Reddit Cross-Topic Authorship Verification Corpus.

Excerpt from original text written by "backpackwayne":

> I am just imagining the debate between Hillary and Rubio. My god she would wipe the floor with him. I mean he might end up crying. The Sanders/Kasich debate would be more competitive but I think Sanders would prevail. I would like nothing more than to see these match ups. I think a Hillary/Sanders ticket would be unstoppable. That does sound like a mess. I can't see how this can be acceptable for a year. There are some important cases in the coming year that will surely end up with this fate. Thanks for clearing that up. Thank you for the information. I was confused on how that would work. :D Say the case was to declare a gay marriage ban unconstitutional. The lower court ruled it unconstitutional. So it would illegal to have a ban in just in just that district. So if gay marriage advocates wanted to have that decision made for the entire country, would they have to start over again with another case, in another circuit. Or would they even be allowed to to do that?...

Excerpt from the GPT Mimic of "backpackwayne":

> As the presidential race heats up, the debates between the candidates become more and more important. I can't help but imagine what it would be like to see Hillary Clinton and Marco Rubio go head to head. In my opinion, Clinton would wipe the floor with him. Rubio might even end up crying. On the other hand, a debate between Bernie Sanders and John Kasich would be more competitive, but I still think Sanders would come out on top. I would love to see these matchups happen. In fact, I think a Clinton/Sanders ticket would be unstoppable. Of course, that would be a bit of a mess, but I can't see how this current situation can be acceptable for a year. There are some important cases coming up in the next year that will surely end up with this fate. For example, let's say a case was brought to declare a gay marriage ban unconstitutional. The lower court ruled it unconstitutional, so it would be illegal to have a ban in just that district. But if gay marriage advocates wanted to have that decision made for the entire country, they would have to start over again with another case in another circuit . . .

We have highlighted in green an example of a sentiment that is the same between the author and its mimic. In yellow are examples of similar writing styles, where both use a forward slash between two last names. Finally, we have highlighted in blue a section of text that the mimic appears to have nearly completely copied from the original author. We point this out as this may cause our model to struggle with detecting the difference between the author and its mimic.

3.3 Modifications to Contriever Model

Developed by Facebook Research, the Contriever model is a document retrieval model based on the large language model BERT [5]. The Contriever model takes a sequence of text, which has been converted into tokens, and generates a feature-rich embedding. The base model is pre-trained on CC-net and English Wikipedia without any supervised data to match queries with documents that likely contain the query's answer. Due to this training, the embeddings generated by this model represent the content of the passage used as input. While the content of a passage is not directly useful for authorship verification, we believe that finetuning can be performed to leverage the pre-trained model's understanding of language, sentence structure, and more for the purposes of authorship verification.

The learning algorithm used for Contriever is known as contrastive learning. As shown in Figure 1, contrastive learning works by adjusting the predictions for positive and negative pairs with respect to an anchor point. For Contriever, the anchor points are queries while the positive pairs are any documents containing the query's answer and negative pairs are any other documents. To apply this learning paradigm to authorship verification, we can set positive pairs as passages written by the same author and negative pairs as passages written by different authors, as shown in Figure 2. By

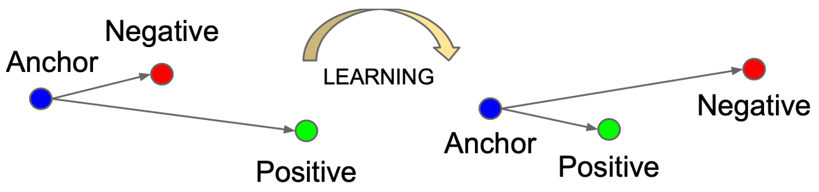

Figure 1: Visualization of contrastive learning with one positive pair and one negative pair.

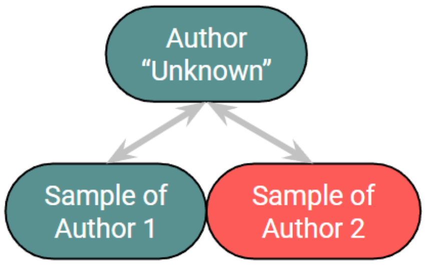

Figure 2: Visualization of passage pairing. The green passages are written by the same author, known as a positive pair. The red passage is written by a different author, known as a negative pair with the "unknown" passage.

applying contrastive learning to this setup, the embeddings will theoretically begin to represent the author that wrote the passage.

To convert the Contriever codebase to work with the Reddit dataset and authorship verification goals, the data set and loader must be adjusted. Since the dataset contains 4 passages for each author known to be written by the author as well as a ChatGPT mimic passage, we can split the dataset into a train set and a validation set. The train set will consist of the first 3 known passages for each author while the validation set will contain the remaining 4th known passage and the ChatGPT mimic passage; however, the ChatGPT author we added does not have a mimic passage for obvious reasons. The dataset contains 1000 authors plus our added ChatGPT author, meaning there are 3003 passages for training and 2001 passages for validation.

While the number of passages is relatively small for a large language model, the number of possible sets of anchor, positive, and negative pairs is much larger. Each set of data contains two random passages from the chosen author, one for the anchor and one for the positive pairs, as well as a random passage from any of the other 1000 authors, for the negative pair. This selection method means that for each author there are 6 possible choices for the anchor and positive pair and 3000 choices for the negative pair. This allows for the model to not overfit on the dataset despite the limited data.

This dataset can then be directly applied to the pre-trained Contriever model for fine-tuning. While training the model on the new dataset, the data loader will combine multiple data points into a single batch and evaluate them in a single iteration. An unfortunate consequence of this is that while all anchor and positive points will be different due to coming from specified authors, the negative points come from any author other than the author for the associated anchor point. Thus a negative point may be written by the same author that wrote the anchor and positive points for one of the other sets of data. It is even possible that a negative point is the exact same passage as the anchor or positive point in another set of data within the batch. The odds of one of these events happening in a given batch is $1-(1-\frac{B-1}{N-1})^B$ where B is the batch size and N is the number of authors. For our training, we use a batch size of 20, and our dataset has 1001 authors meaning the odds of a negative sample sharing an author with an unrelated anchor and positive point in the batch is $1-(1-\frac{20-1}{1001-1})^{20}=0.319$. While this event is reasonably likely to occur in any batch, we have chosen to allow these rare events in hopes that the training will be able to handle these cases.

4 Results

After fine-tuning the Contriever model on the Reddit dataset with the ChatGPT addition, we applied the trained model to every passage in the dataset to generate normalized embeddings for all of the data. Using these embeddings, we can evaluate the performance through various metrics.

4.1 One vs One Comparisons

One method for evaluating the performance of authorship verification is by performing tests between two authors: given a single sample from each of the two authors and a third sample from an unknown author, can the model reliably identify the correct author. This evaluation setup is shown in Figure 2. We choose the known samples from the validation set of two random authors and the unknown sample from the training set of one of the chosen authors. The author with the higher score, calculated by taking the dot product of the embeddings for the known and unknown passages, is determined to be the predicted author of the unknown passage. By performing this test 10,000 times, the model is shown to have an accuracy of 69.54% which is greater than the random chance of 50% but not significantly.

Rather than only utilizing one known sample for each author, we can instead use all 3 training samples from each author as the known passages and the validation passage from one of the two authors as the unknown passage. By calculating the scores for the pairings of each known passage with the unknown, the author which has the highest of all scores is designated as the predicted author of the unknown passage. This evaluation method results in an accuracy of 76.76% after 10,000 tests. While this is an improvement on the previous metric, the accuracy is still below other authorship verification methods.

4.2 Full Set Comparisons

The purpose of developing this model revolves around testing for dishonesty in authorship claims, specifically with ChatGPT as a method for writing documents. Despite the purpose being focused on choosing between a supposed author and a suspected author, it is also interesting to look at how well the model performs when the entire dataset of authors are listed as possibilities.

When pairing the training set data against itself and removing any self-pairings, we can evaluate how many passages are ranked as a higher pair for a given passage before a passage written by the same author appears. Since there are 3002 passages in the train set that can be paired with any chosen passage with 2 of those passages being written by the same author, a random sampler would achieve a first choice accuracy of $\frac{2}{3002} = 0.067\%$. Our model greatly exceeds this with a first-choice accuracy of 2.33%.

Rather than pairing the training set against itself, we can pair the validation set against the train set. In doing so, we can rank the passages from the training set based on their score with a given validation passage. The first-choice accuracy for a random sampler would be $\frac{3}{3003} = 0.1\%$ while our model reaches 2.50%. Looking beyond the first choice, the model is able to find a passage written by the correct author within 5 passages 6.79% of the time, within 20 passages 17.48% of the time, and within 100 passages 40.46% of the time. Plotting this data for all possible numbers of passages results in Figure 3 which shows results significantly higher than random selection.

Similar to the method used in the One vs One Comparison section to increase accuracy, we can combine the train set into average embeddings for each author. By pairing the evaluation set with the averaged train set, we find that the first choice accuracy jumps up to 4.90% when the expected is only $\frac{1}{1001} = 0.1\%$, the same as before. Also, the model is able to find the correct author within 5 authors 11.09% of the time, within 20 authors 19.88% of the time, and within 100 authors 43.66% of the time. One thing to note is that there is only 1001 possible authors this time compared to the 3003 possible passages in the prior evaluation. When plotting this accuracy as a function of the number of predictions, Figure 4 shows a significant increase over random.

4.3 ChatGPT Mimic

The final evaluation we ran compared the ChatGPT mimics with the respective mimicked author and the ChatGPT author we added to the Reddit Cross-Topic Authorship Verification Corpus. When

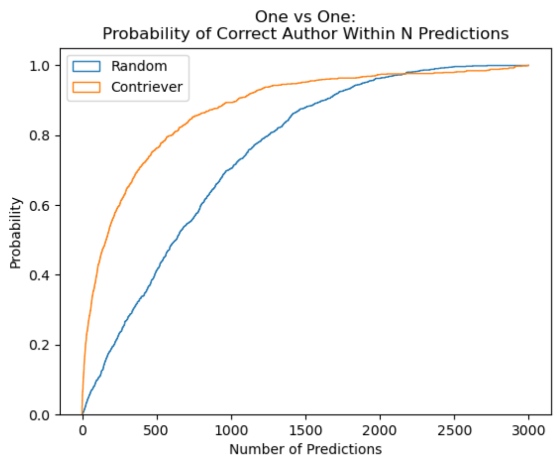

Figure 3: Probability of pairing an unknown passage with a passage written by the correct author within N pairings.

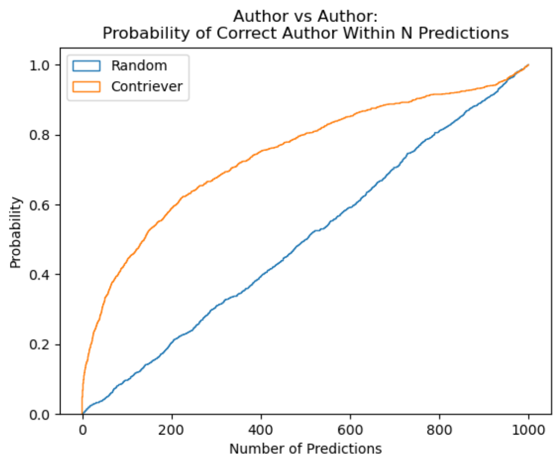

Figure 4: Probability of pairing an unknown passage with the correct author within N pairings.

using one training sample from the mimicked author as the negative pair and ChatGPT as the positive pair as in the One vs One Comparisons, the model correctly identifies the mimic passage as written by ChatGPT 19.70% of the time. When using all 3 training samples from the mimicked author and ChatGPT, the accuracy actually drops to 13.93%. These results showcase that our model is being fooled by ChatGPT trying to mimic an author. It is important to note that the model is not trained on any mimic data which could allow for the model to learn mistakes made by ChatGPT while mimicking authors.

5 Discussion and Future Work

Unfortunately, our Contriever model was unsuccessful in detecting the ChatGPT mimicked text. While our model did see success in identifying authors without ChatGPT involved at approximately 70% accuracy, the accuracy in determining mimicked authors was between approximately 14% and 20%. This accuracy is not useful in detecting ChatGPT as it is worse than a random guess. We investigated a few potential causes for this inability of our Contriever model to detect ChatGPT.

During our analysis of the data used to evaluate our model, in some of the mimicked text outputs, the output directly matches or references some of the author's text that ChatGPT is attempting to mimic. The copying of specific phrases of the text ensures that ChatGPT's writing samples appear very close to the author. This likely makes it more difficult to determine if the writing sample is ChatGPT or the original author. Future work in modifying prompts to have a better, more unique writings by ChatGPT on new topics rather than just reincorporating the original author may prove to have more success in detection. This would also provide a more realistic use case for detection. Empirically, most cases where ChatGPT abuse occurs is when it is generating new text, rather than text based on already existing responses. A student attempting to have ChatGPT write an essay based on their past essays for example will not write the essay first, and have ChatGPT rewrite it in their style. As such, further investigation in getting ChatGPT to generate new and unique discussions on topics based on the writing style established in a given piece of text would likely improve our results.

Additionally, this may be an issue based on the Contriever model itself, which was originally designed for document retrieval. The use of an already existing model developed for the problem of authorship verification may achieve better results in detection.

While our model did not achieve success in detecting ChatGPT mimicking authors, we have created a useful test bed of data for future investigation. Additionally, we identified several promising ways to improve upon our model through the modification of ChatGPT prompts and the utilization of more authorship verification techniques.

References

[1] Nash Anderson, Daniel L Belavy, Stephen M Perle, Sharief Hendricks, Luiz Hespanhol, Evert Verhagen, and Aamir R Memon. Ai did not write this manuscript, or did it? can we trick the ai text detector into generated texts? the potential future of chatgpt and ai in sports & exercise medicine manuscript generation. *BMJ Open Sport & Exercise Medicine*, 9(1), 2023.

[2] Evan Crothers, Nathalie Japkowicz, and Herna Viktor. Machine generated text: A comprehensive survey of threat models and detection methods, 2023.

[3] Mendeley Data. Enron authorship verification corpus. `https://data.mendeley.com/datasets/hppkn5kbg8/1`, 2016.

[4] Mendeley Data. 400 compositions by efl learners of a university in southwestern china (cefllusc). `https://data.mendeley.com/datasets/jxm2bwkc33/1`, 2022.

[5] Gautier Izacard et al. Unsupervised dense information retrieval with contrastive learning, 2021.

[6] OpenAI. Introducing chatgpt. `https://openai.com/blog/chatgpt`, 2022. Accessed: 2023-03-24.

[7] Raf. What are tokens and how to count them?: Openai help center.

[8] Teo Susnjak. Chatgpt: The end of online exam integrity?, 2022.