Detection of AI-generated text and Query guessing

Abhimanyu Bellam, Vishnu Vinod Erapalli, Ankur Banerji, Harshwardhan Joshi
North Carolina State University, Raleigh, USA
Email: abellam2@ncsu.edu, verapal@ncsu.edu, abaner24@ncsu.edu, hjoshi2@ncsu.edu

I. Introduction and Background

A. Problem statement

1) The general problem: AI is revolutionizing technology and providing breakthroughs in every aspect of our lives. It was crafted to help automate and simplify repetitive tasks whilst also possessing intelligence of its own. Learning from data, models have the capability to use the same when presented with new and unseen data. AI also possesses the ability to generate text of its own, given a topic and it also does it within the specified word limit. This has also been rampant in the field of academia as well. As much as students use their own abilities and knowledge, they resort to online tools and aid to ease their academic workload. In today's day and age, ChatGPT has taken over and its widespread use has left the world awestruck.

Academia requires individuals to come up with some text of their own, based on their understanding. Using AI to generate text is considered an act of academic dishonesty, and is becoming a widespread problem. A method to check if some text was AI-generated would be of great use in today's time, especially for professors and teachers. This has led to people developing AI-based models to identify AI-generated text.

2) The specific Problem: Using state-of-the-art deep learning techniques such as BERT and BART [1], we intend to identify whether or not a given piece of text is human-written or AI-generated via the text summaries we acquire from the datasets.

We perform text data exploration to analyze the structure, format, and parts of speech of Human and AI-written texts, build a text classification model which detects if a text is AI-generated or Human written. We will also explore robustness by checking if the model gives the same results when the common words are removed, check similarities between summaries output by summarization models on AI-generated text and Human written text. Further, we intend to perform exploratory analysis by supplying "extra information" to the AI tool to generate new text and pass this to our text classifier to check if it would still be able to detect if the text was AI or human generated. Additionally, we try to remove 'hard words' or 'difficult-words' from the dataset and pass the same to our classifier to check whether or not it was able to successfully detect who the writer was, i.e, AI or Human. The backing to this experiment is that usually, students tend to pass a topic to AI and get an AI-generated response. To make this less obvious, they try to plant in some 'hard words' and make it appear as if they themselves have written the text.

B. Related work

There were numerous articles and resources that we drew inspiration from. Most articles explained strategies on a high level and this helped us channel our thoughts. We spent time reading about existing and proposed methods to shape our understanding about the topic. The first resource was a research paper titled 'Automatic Detection of Generated Text is Easiest when Humans are Fooled' [2] where the authors researched about detecting machine-generated text. They tried to compare human detection methods vs automatic methods. They used benchmarking and analysis of three popular sampling-based decoding strategies— took, nucleus sampling, and untruncated random sampling. Although most of the research was fairly complex to understand, this paper gave us a direction of thought and it also made it clear that when humans are involved, there is more play concerning semantics. Our next resource was the OpenAI website [3] where they talked about their classifier which could detect AI-generated texts. We learned that they divided each text into a prompt and a response. On these prompts, responses were generated from a variety of different language models trained by them and other organizations. They adjusted the confidence threshold to keep the false positive rate low, i.e, they only mark text as 'likely AI-written' if the classifier is very confident. The third resource was a research paper titled 'Can AI-Generated Text be Reliably Detected?'[4] and this helped us understand that detectors can be fooled. The detectors tend to use certain techniques like watermarking techniques and this can allow us to find workarounds by using light paraphrasing tools. The next resource was 'On the Possibilities of AI-Generated Text Detection' [5]. The authors conducted experiments on IMDb dataset to support their assertions regarding the viability of detection. Their dataset comprised of 50,000 movie reviews and this is relevant to us since we too are processing a huge dataset. They then talked about refining their model and using it at a paragraph level and they achieved a train AUROC of greater than 0.85, and a test AUROC of greater than 0.8 which surpasses the upper bounds of the best detector. From this paper, it was inferred that a detector would work much better when used on a paragraph-level as compared to word-level. Last but not the least, we referred to 'Stylometric Detection of AI-Generated Text in Twitter Timelines' [6]. The authors used stylometric traits (Phraseology, Punctuation, and Linguistic Diversity) in their methodology as an auxiliary signal to supplement the available SOTA detectors.Their suggested fusion network combines stylometric information with

PLM's semantic embedding capabilities. For stylometry fusion architecture in the job of human- vs. artificial intelligence (AI)-authored tweet detection, the authors decided to employ RoBERTa as the language model. This research paper allowed us to understand that classifying small texts as AI or human was a near-impossible task, as even the most state-of-the-art detection algorithms struggle with it. Hence, we decided that using longer texts was the best approach, as AI-generated mistakes tend to creep out in longer texts, as pointed out by the authors of this paper. Overall, the research material and online resources opened our eyes to various techniques out there and we came up with an approach that we feel is novel and would tackle the use case in a fairly accurate manner.

II. Method

A. Novel Aspect(s)

1) There are many Text classification models but our focus is on detecting AI-generated texts which makes the task all the more challenging. Thus, for this task, the BERT model was used and this was not deployed directly but was retrained on our dataset in batches. The dataset was computationally expensive and hence was divided into batches. There are many hyperparameters involved and finding the right balance becomes very crucial in such cases. Since the training on such a huge dataset is time-consuming and each epoch takes about 105 minutes, Grid search becomes very strenuous. Therefore, based on research and acquired knowledge, we chose an optimal set of hyperparameters that helped us achieve high accuracy.
2) Not only do we explore if the text is AI-generated or not, but we go beyond this question and we try to explore whether eliminating hard words affects the model's recognizing capability. Hard words and English Language analysis is another aspect we explore and we perform exploratory data analysis on this. Additionally, we trained a model 'BART' which yielded text summaries of the data generated by AI and Humans. We checked for similarity in the summaries by writing a cosine similarity function that processed the summaries batch-wise and returned the number of responses that showed higher similarity. This was later passed to our model to check if the model performed well in detected AI-generated text.
3) Based on the performance of the model on the given original dataset(i), the dataset with removed 'hard words'(ii), and the generated summaries of the dataset(iii), we try to explain the displayed behavior of the model. This will enable us to better understand the data and the model and what can be done to improve the performance of the model. This will also raise the question that if our chosen model is the right one for this use case or not.

B. Rationale

Some of the traditional baseline models for text classification are: logistic regression where the texts might be encoded to vectors, naive bayes classifier which is a probabilistic model that assumes that the occurrence of a particular word in a document is independent of the occurrence of other words, and random forest which is a flexible and robust model that can handle high-dimensional input data, making it a useful baseline model for text classification tasks with many features. We use BERT as a text classification model because when it comes to detecting AI-generated text, BERT is a highly sophisticated and powerful pre-trained model that is capable of capturing the nuances of language and context. This is important when dealing with AI-generated text that may be designed to mimic human language patterns. This model has mechanisms that allow it to capture the context and meaning of each word in a sentence. This can be crucial in detecting subtle differences in Human and AI-generated texts.

Moreover, we go beyond the baseline models in the sense that, we also experiment with feeding different variants of data which also encompass summaries and hard-words removed datasets. To generate the Summaries we chose BART since it is a modern text summary that we do not require to be fine-tuned to our data. The model's input is text and is encoded into a high dimensional encoding by a bidirectional encoder and decoded by an autoregressive decoder (next word is dependent on the previous word) to generate the output. In our case, we need to maintain the flow of text and we need this feature where each word is dependent on the previous word since this helps maintain the meaning of the group of words because they are not merely stray words existing together. They make sense as they occur in a sentence.

To find the similarity in generated summaries, we encode them into vectors and use cosine similarity. This similarity measure works best since it is: 1.Scale-invariant,i.e, Cosine similarity is not affected by the length of the documents being compared, 2.Directional: Cosine similarity considers the direction of the vectors rather than their magnitude. This means that two summaries with similar content but different lengths will still have a high cosine similarity score, 3.Efficient: Cosine similarity can be computed quickly and efficiently, even for large datasets, and considering the fact that our dataset is huge, this makes measure makes the most sense.

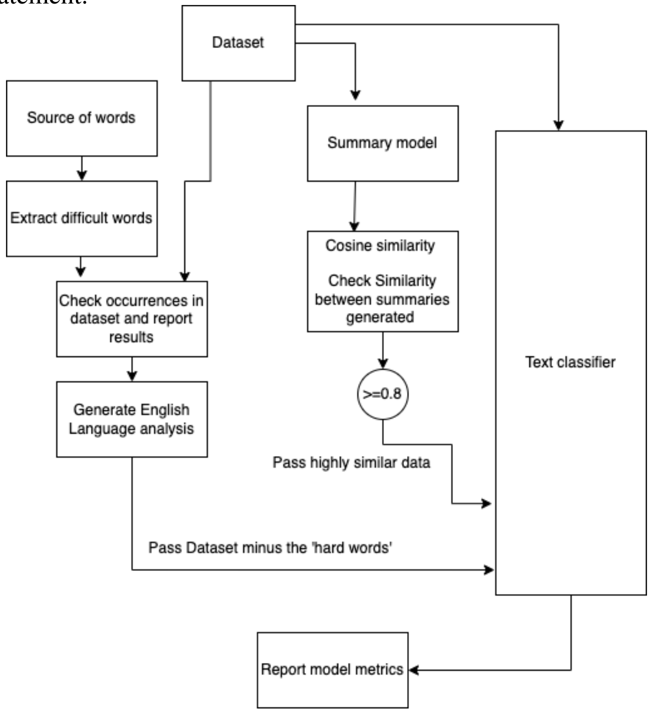
C. Approach

We use the GPT wiki-intro Dataset which has human-written and AI-generated introductions for 150,000 Wikipedia topics and this dataset is first passed to the BART model. This is a pre-trained sequence-to-sequence model that can be fine-tuned for various downstream natural language processing (NLP) tasks such as text summarization, text generation, and question-answering. Since the dataset is huge, the generated summaries are uploaded in batches. The summaries generated are then taken and their similarity is checked. This tells us how similar the summaries of Human-generated and AI-generated texts are. For this cosine similarity is used and we use a threshold of 0.8. All the rows that have a similarity higher than this are selected and passed to the text classification model.

Parallelly, we explore words that are more commonly used by humans. These words are called 'Hard words' and by experience, it is determined that AI does not commonly generate these words. The difficult words are obtained and the occurrences of these are checked for in our dataset. Moreover, we perform an English language analysis that checks for the occurrences of common parts of speech such as nouns, verbs, stop words, and adverbs in each of the two generated texts. The dataset is then modified to eliminate the hard words and this is passed to our Text classification model.

The core implementation of our project revolves around the text classification model and for this we use the BERT model. This model is re-trained on our dataset. We use adversarial training, where BERT is trained on a combination of human and AI-generated texts and is encouraged to identify the differences between them. Once this is done, we pass the dataset without hard words and check if it is able to differentiate between AI or humans generated texts. Next, we feed in the text summaries with high similarity to check if the model performs well.

The following flowchart shows our approach to the problem statement.


Dataset
Source of words
Summary model
Extract difficult words
Cosine similarity
Check Similarity between summaries generated
Check occurrences in dataset and report results
Text classifier
>=0.8
Generate English Language analysis
Pass highly similar data
Pass Dataset minus the 'hard words'
Report model metrics

III. Plan and Experiment

A. Datasets

We plan to use the GPT-wiki-intro Dataset which has Human-written and GPT (Curie) generated introductions for 150,000 Wikipedia topics. The GPT-wiki-intro dataset hosted on Hugging Face is a dataset of Wikipedia introductory paragraphs, preprocessed and tokenized for use with the GPT language model. The dataset contains over 6 million paragraphs from a variety of Wikipedia articles, covering a broad range of topics.

Each paragraph in the dataset is tokenized using the GPT tokenizer, which splits the text into tokens (words and punctuation) and assigns a numerical ID to each token. The purpose of this dataset is to provide a large, high-quality corpus of text for use with the GPT language model, which is a powerful tool for natural language processing tasks such as text generation, question answering, and language translation.

The dataset consists of 12 columns, namely, id, URL ,title, wiki-intro, generated-intro, title-len, wiki-intro-len, generated-intro-len, prompt, generated-text, prompt-token and generated-text-tokens. However, while training our AI-generated text classifier model, we plan on using wiki-intro and generated-intro. To train the model which guesses the prompt, we plan on using the generated intro and prompt.

B. Hypothesis

Given that our text classifier performs well on the GPT Wiki intro dataset, we hypothesize that:
(i) The model performs not so well on the dataset with 'hard words' removed
(ii)The model performs equally badly when summaries with high cosine similarity are passed to the model.
(iii) the AI-generated text summaries and Human-generated text summaries outputted by our summary model show less similarity with each other.

C. Experimental Design

In the quest for dealing with the first hypothesis, we should first perform English analysis and extract the difficult words.

1) English Language and Word Analysis: Exploratory data analysis of the dataset provided some vital information about the behavior of AI-generated texts. The first observation from the dataset tells us that for AI to generate some text, it requires more than just the topic of said text. In the given dataset, this extra bit of information is provided in the "prompt" attribute, which specifies the type of text required, and provides a cue for the AI to begin generating text.

A very important feature of the dataset is the word count, which reveals that human-generated texts ("wiki-intro column) are generally much longer than AI-generated texts ("generated-intro" column). Analyzing the word count of the dataset revealed that, on average, human-generated texts have 195.53 words, whereas AI-generated texts have 131.01 words. Additionally, human-generated texts ranged between 35 and 447 words, while human-generated texts ranged between 7 and 282 words. One of the reasons for this disparity in word count can be attributed to the fact that AI-generated texts are typically based on statistical patterns and algorithms. Since models like GPT-3 are limited to the data they are trained on, they are unlikely to produce nuanced and complex ideas and theories, which result in shorter text generation. Further, AI-generated texts are optimized to deliver information concisely, which means that such texts have emphasis on brevity and efficiency.

This ultimately leads to shorter texts that prioritize information delivery.

We have used the Natural Language Toolkit (NLTK) for this. NLTK contains a corpus of stop words which can be used to eliminate the said low level information from the dataset. All kinds of analysis performed on the dataset from this point is done after removing these stop words.

The next step in the analysis of the data is text pre-processing. It involves preparing text data for the machine to be used for predicting, analysing, etc. We use Natural Language Processing (NLP) tools to perform text pre-processing. Removing ”stop words” is the first step in text pre-processing. Stop words are those that are typically excluded from natural language processing. The text does not get much information from these terms, which are among the most common in any language (along with articles, prepositions, pronouns, conjunctions, etc.). Some examples of stop words in English are ”the,” ”a,” ”an,” ”so,” and ”what.” By getting rid of these words, we can make our text more focused on the key information by eliminating the low-level information.

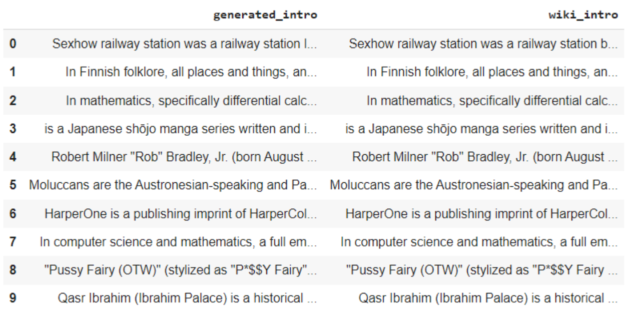
Next, we have compiled a list of 5,350 ”difficult” words, i.e., strong words in English that are not commonly used. These words are used to describe the ”vocabulary strength” of some texts. We have used this metric to identify if stronger words are used more by humans or by AI. A strength function was applied to both the ”wiki-intro” column and ”generated-intro” column to get the said result, and it was observed that human-generated texts contain more of these ”difficult” words than AI-generated texts. A total of 434,891 difficult words were observed in the ”wiki-intro” column, whereas 275,800 difficult words were observed in the ”generated-wiki” column, which is almost half the number found in ”wiki-intro”. Another interesting observation made was that on average, human-generated texts contained 2.92 diffiuclt words, while AI-generated texts contained 1.86 difficult words. This observation can once again be attributed to the fact mentioned above. The use of literary devices such as metaphors, allusions, and the fact that humans can draw from a wide range of experiences contributes to the fact that stronger vocabulary is found mostly in human-generated texts.

2) Text classification model: For this, we used an 80-20 split for train and testing and this 80 has been used for 5-fold cross-validation.

Text classification requires certain packages including NLP tools. A pre-trained transformer-based neural network model called DistilBertForSequenceClassification is employed for text classification tasks. It is a distilled version of BERT (Bidirectional Encoder Representations from Transformers), which means it has been condensed and streamlined for quicker and more effective inference.

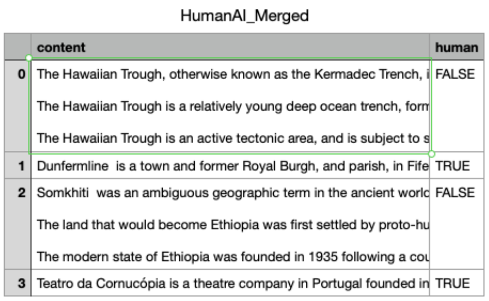
For sequence classification applications like sentiment analysis or document classification, the DistilBertForSequenceClassification model was created particularly. In a labeled dataset of examples.

The dataset, i.e. ’GPT-Wiki-Intro’ has two columns that the classification task is concerned with. The ’generated-intro’ speaks about the textual data that is generated by a bot i.e. AI essentially, and the second column i.e. ’wiki-intro’ is generated by humans. The classifier explained later, will learn the relationship between the texts generated by humans and the same done by AI through these two columns. A view of the dataset of the relevant portion is shown below:



	generated_intro	wiki_intro
0	Sexhow railway station was a railway station l...	Sexhow railway station was a railway station b...
1	In Finnish folklore, all places and things, an...	In Finnish folklore, all places and things, an...
2	In mathematics, specifically differential calc...	In mathematics, specifically differential calc...
3	is a Japanese shōjo manga series written and i...	is a Japanese shōjo manga series written and i...
4	Robert Milner "Rob" Bradley, Jr. (born August ...	Robert Milner "Rob" Bradley, Jr. (born August ...
5	Moluccans are the Austronesian-speaking and Pa...	Moluccans are the Austronesian-speaking and Pa...
6	HarperOne is a publishing imprint of HarperCol...	HarperOne is a publishing imprint of HarperCol...
7	In computer science and mathematics, a full em...	In computer science and mathematics, a full em...
8	"Pussy Fairy (OTW)" (stylized as "P*$$Y Fairy"...	"Pussy Fairy (OTW)" (stylized as "P*$$Y Fairy ...
9	Qasr Ibrahim (Ibrahim Palace) is a historical ...	Qasr Ibrahim (Ibrahim Palace) is a historical ...

Amalgamation of the dataset takes place by combining both the columns of the dataset into one column and shuffling them to randomly place rows with AI-generated and human-generated text together. Additionally, a column with the label ’Human’ is added to indicate whether any specific row is human-generated with two values - True and False. The total number of data points that we are working with currently is 300K data points. The splitting of the dataset is done with the training set receiving 70% of the actual dataset i.e. 210K data points with a random seed to reduce the probability of the imbalance of the two categories. The validation set receives 15% of the data i.e. 45K data points. Finally, the testing data contains the rest of the data points.



HumanAI_Merged

	content	human
0	The Hawaiian Trough, otherwise known as the Kermadec Trench, i The Hawaiian Trough is a relatively young deep ocean trench, forn The Hawaiian Trough is an active tectonic area, and is subject to s	FALSE
1	Dunfermline is a town and former Royal Burgh, and parish, in Fife	TRUE
2	Somkhiti was an ambiguous geographic term in the ancient worlc The land that would become Ethiopia was first settled by proto-hu The modern state of Ethiopia was founded in 1935 following a cou	FALSE
3	Teatro da Cornucópia is a theatre company in Portugal founded in	TRUE

The texts need to be pre-processed. The first step of any text pre-processing starts with tokenizing the sentences. A tokenizer, which separates text into tokens or subwords, is an element in natural language processing (NLP) pipelines. We use DistilBertTokenizerFast to do the same.

We created a class to combine the encodings and the labels so that we can pass it into pyTorch’s dataloader. We made sure that the training data is shuffled.

The hyperparameters used are:
(1) Batch Size: 16
We wanted give sufficient attention to fluctuations and varaitions in data. Thus we went for a slightly lower batch size with respect to the overall data size. However, we would not go lower than 16 as it would affect the overall learning of the network.
(2)Warm-up steps: 500

This hyperparameter helps the network adapt to the data and to support the optimizers to perform correct statistics.

The number of epochs used were 3.

3) Text summarizer and setup: Our setup is quite compute intensive in the fact that we train a heavy classifier for our purpose and that we use a summarizer to get text summaries. Therefore, we created a VCL (NC State Virtual Computing lab) setup with shell script for installations and auto downloads for easily running our classifier and text summarizer multiple times due to the 8hr restriction of the VCL image. We leveraged the RTX 2080 Ti GPU image to perform our experiments. As part of trying to answer if the question, "If you summarized an AI-generated text and a Human generated text on the same topic, would you get similar summaries?", we used a modern text summarizer called BART (Bidirectional Auto-Regressive Transformers) [1]. The model's input is text, which is encoded into a high dimensional encoding by a bi-directional encoder and decoded by an autoregressive (next word is dependent on the previous word) decoder to generate the output. Our experiment does not require BART to be fine-tuned to our use case because we intend to use the summary as a tool to verify the example (input) difficulty for our classifier. BART summarization involves tokenizing and encoding the inputs and decoding the output. We generated summaries for each of the 150,000 AI texts and 150,000 human texts.

4) Check for Cosine Similarity of Summaries Generated: Two columns of the CSV selected, i.e, wiki-intro and generated-intro were taken and passed to our summary model which resulted in the summaries.

We first had to encode the text. This was done using the embedding functionality of TensorFlow. TensorFlow Embedding is a technique used in machine learning and natural language processing to convert textual data into numerical vectors that can be understood by a neural network. It maps words or phrases to a high-dimensional vector space, where similar words are located closer together. TensorFlow Embedding works by creating a lookup table that assigns a unique vector to each word in a vocabulary. The embedding vectors are learned during the training of a neural network, and their values are adjusted based on the model's ability to predict the target output. By using embeddings, the neural network can learn to understand the meaning of words and phrases and can make more accurate predictions based on this understanding.

The encodings are done by taking one column at a time. The resulting encoding is of the shape and format TensorShape([90000, 512]). Each summary/ row is encoded as a 1x512 vector. We have 90000 rows since we filter only those rows from the original dataset which was used by our text classification model for its validation and testing.

In order to perform easier computations and processing, the tensor is converted into an Nd array. This involves an intermediate step that uses the make-tensor-proto function of tensor flow. The tf.make-tensor-proto function takes a tensor object and returns a protocol buffer representation of the tensor. The resulting protocol buffer can be used to transmit the tensor over a network, store it on disk, or pass it between different parts of a TensorFlow system. This helps us use the make-nd-array function on the result and the end result is an nd array.

The next stage to finding similarity between two text summaries is the cosine similarity measure. This is one of the fundamental similarity measures which can be used whilst comparing or finding similarities between two vectors.

The task is now to find a similarity if any between the two texts' summaries.

Cosine similarity is used as a similarity measure that calculates the cosine of the angle between two vectors in a high-dimensional space. The resulting value ranges from -1 to 1, where 1 indicates that the vectors are identical, 0 indicates that the vectors are orthogonal (i.e., have no similarity), and -1 indicates that the vectors are diametrically opposed. It is given by cosine-similarity = (a . b) / (//a// * //b//)

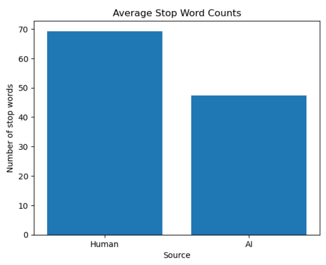
We compute the cosine similarity of resulting vectors and if the similarity crosses 0.8, we record that and increase our count. We track the number of rows that exceed our set threshold of 0.8. We observe that for 45000 testing samples, 860 of them have very similar text summaries, i.e, the summaries generated by our summarization model for AI-generated text and Human-generated text are very similar in 860 instances. Now, out of the 150K rows in our dataset, only 860 of them shows that there is high similarity between the two text summaries. This confirms the hypothesis that AI-generated text summaries are not similar to Human-generated ones since 860 is not statistically significant in our case and it might have arisen by chance.

IV. Results

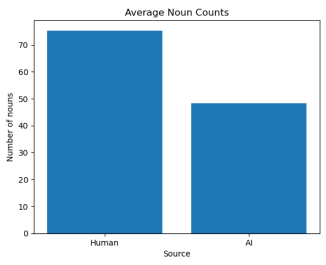
A. English language analysis

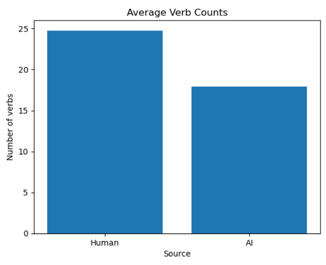
Analysis of text data was performed using NLTK. We have focused our analysis on the main parts of speech: nouns, verbs, adjectives, adverbs, and pronouns. Apart from this, we have also analyzed the usage of stop words in both kinds of texts, in order to get a better understanding of the differences between AI-generated and human-written texts. NLTK also provides a POS(part-of-speech) tagging that is used to label each word to its corresponding part of speech. We have the following tags to label words: N - Nouns, V - Verbs, J - Adjectives, R - Adverbs, P - Pronouns. Some important observations from our analysis have been highlighted below:

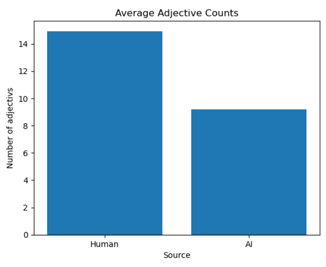
1) Stop words analysis: From the graph below, it can be observed that on average, human-written texts contain more stop words than AI-generated texts. Human texts use an average of 69.7 stop words per text, while AI-generated texts use 48.9 stop words.

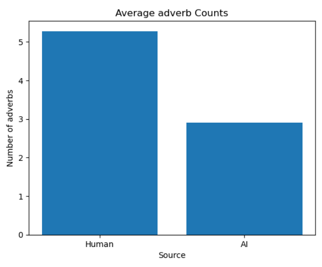


2) Parts of speech analysis: Using the above mentioned POS Tagging, we have generated the following graphs to observe counts of different parts of speech.
1. Nouns - Average nouns in human-written texts = 74.8, average nouns in AI-generated texts = 47.1.
2. Verbs - Average nouns in human-written texts = 24.7, average nouns in AI-generated texts = 17.1
3. Adjectives - Average adjectives in human-written texts = 15.3, average nouns in AI-generated texts = 9.4
4. Adverbs - Average adverbs in human-written texts = 5.2, average adverbs in AI-generated texts = 2.8





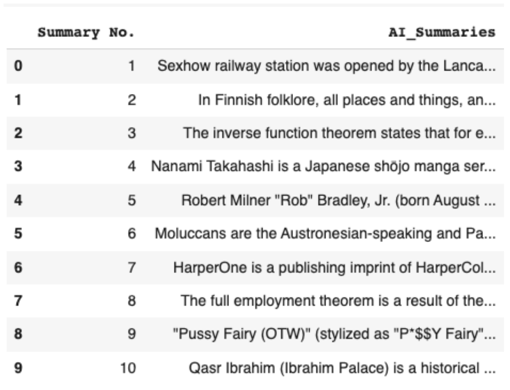




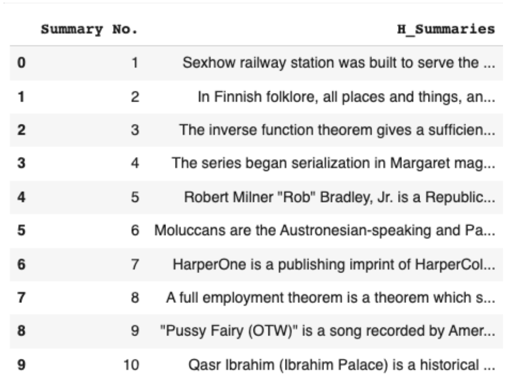
Based on the English language analysis we can see that with respect to the parts of speech, human-written texts have more occurrences of each of them. This can be attributed mostly to the fact that human-written texts are longer than AI-generated texts (i.e., the average word-count of human-written texts is more than the average word-count of AI-generated texts).

B. Summarization model

The summaries generated by the BART model are concise and they encapsulate the idea of the text. This is true in both the cases of human and Ai generated texts. There is no inherent difference between the two.



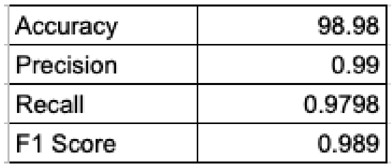
	Summary No.	AI_Summaries
0	1	Sexhow railway station was opened by the Lanca...
1	2	In Finnish folklore, all places and things, an...
2	3	The inverse function theorem states that for e...
3	4	Nanami Takahashi is a Japanese shōjo manga ser...
4	5	Robert Milner "Rob" Bradley, Jr. (born August ...
5	6	Moluccans are the Austronesian-speaking and Pa...
6	7	HarperOne is a publishing imprint of HarperCol...
7	8	The full employment theorem is a result of the...
8	9	"Pussy Fairy (OTW)" (stylized as "P*$$Y Fairy"...
9	10	Qasr Ibrahim (Ibrahim Palace) is a historical ...



	Summary No.	H_Summaries
0	1	Sexhow railway station was built to serve the ...
1	2	In Finnish folklore, all places and things, an...
2	3	The inverse function theorem gives a sufficien...
3	4	The series began serialization in Margaret mag...
4	5	Robert Milner "Rob" Bradley, Jr. is a Republic...
5	6	Moluccans are the Austronesian-speaking and Pa...
6	7	HarperOne is a publishing imprint of HarperCol...
7	8	A full employment theorem is a theorem which s...
8	9	"Pussy Fairy (OTW)" is a song recorded by Amer...
9	10	Qasr Ibrahim (Ibrahim Palace) is a historical ...

C. Text classification model

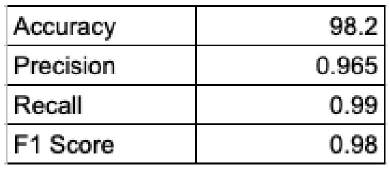
It is seen that the model does well on the dataset we provided because the BERT model has already been trained on the wiki intro dataset and thus it has seen enough samples to do well on the test data. BERT model has been already trained on a huge dataset of 3.3 billion words which is a combination of Book Corpus (800M words), English Wikipedia (2,500M words), and a subset of the English version of the Common Crawl (76M words). Moreover, we have used cross-validation to get the best model on that data. The metrics are shown here:



Accuracy	98.98
Precision	0.99
Recall	0.9798
F1 Score	0.989

We comment on the performance based on accuracy. The higher precision and recall scores are due to the fact that it has seen enough Wikipedia data already in order to correctly differentiate between AI and Human-generated texts. This may not be the case in real-life scenarios because a student's piece of writing may not actually be written like a Wikipedia article which we here feed to our model.

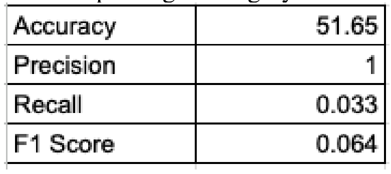
When we remove the difficult words from the dataset:



Accuracy	98.2
Precision	0.965
Recall	0.99
F1 Score	0.98

We observe a slight decrease in accuracy and a fall in precision. The recall is slightly higher and the combination of these values suggests that false positives increased and false negatives decreased. This is somewhat in accordance with our hypothesis that the model will perform worse when we remove the hard words.

After passing the highly similar summaries to the model:



Accuracy	51.65
Precision	1
Recall	0.033
F1 Score	0.064

We see that the model does worse on this. The accuracy is very low and so is the recall and F1 score. However, precision is higher because there are only two classes and it is predicting everything as almost one class. So when TP>>FP, it is the case that TP/(TP+FP).

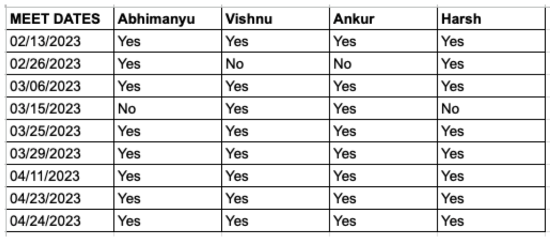
D. Discussion

Based on the formulated hypothesis, our experiment agrees with all of them. Overall, from these experiments, we think that the model is learning well and is not exactly focusing on the usage of hard words. This is evident from the fact that its performance did not drastically drop when hard words were removed from the dataset. The model focused more on language structure and semantics. As hypothesized, the model did badly on the summarized texts that were highly similar because human-generated texts were summarized by AI itself and this would've led to some bias. It could also be the case that human texts are longer and then the model would've learned to generalize that all long texts are human-generated. Prior work in this field saw that they worked on texts on the paragraph level. However, we feel that this does not entirely capture the nuances of the language and thus we perform a word-by-word analysis.

E. Conclusion

With this experiment, we have bolstered our machine learning and data analysis skills. We learned to formulate a viable hypothesis on a relevant use case and we acquired the skills of dealing with models that help achieve these tasks. Moreover, we learned to train models which are challenging and state-of-the-art. During the course of this project, we learned that when dealing with huge datasets, model training becomes very time-consuming. Therefore experimenting on huge datasets is not the most logical thing to do. Dividing data into batches for training is the best approach. We aimed to feed over 150,000 topics to chatGPT but were unable to do so since we did not get our hands on a free version of OpenAI's GPT APIs. We wanted to use this data to actually prove the working of our model.

V. Meeting Schedule



MEET DATES	Abhimanyu	Vishnu	Ankur	Harsh
02/13/2023	Yes	Yes	Yes	Yes
02/26/2023	Yes	No	No	Yes
03/06/2023	Yes	Yes	Yes	Yes
03/15/2023	No	Yes	Yes	No
03/25/2023	Yes	Yes	Yes	Yes
03/29/2023	Yes	Yes	Yes	Yes
04/11/2023	Yes	Yes	Yes	Yes
04/23/2023	Yes	Yes	Yes	Yes
04/24/2023	Yes	Yes	Yes	Yes

VI. Github Repo

https://github.com/AbhimanyuBellam/plagDetector

References

[1] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2019, October 29). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. arXiv.org. https://arxiv.org/abs/1910.13461v1

[2] Ippolito, D., Duckworth, D., Callison-Burch, C., & Eck, D. (2019, November 2). Automatic Detection of Generated Text is Easiest when Humans are Fooled. arXiv.org. https://arxiv.org/abs/1911.00650v2

[3] New AI classifier for indicating AI-written text. (n.d.). New AI Classifier for Indicating AI-written Text. https://openai.com/blog/new-ai-classifier-for-indicating-ai-written-text

[4] Sadasivan, V. S., Kumar, A., Balasubramanian, S., Wang, W., & Feizi, S. (2023, March 17). Can AI-Generated Text be Reliably Detected? arXiv.org. https://arxiv.org/abs/2303.11156v1

[5] Chakraborty, S., Bedi, A. S., Zhu, S., An, B., Manocha, D., Huang, F. (2023, April 10). On the Possibilities of AI-Generated Text Detection. arXiv.org. https://arxiv.org/abs/2304.04736v1

[6] Kumarage, T., Garland, J., Bhattacharjee, A., Trapeznikov, K., Ruston, S., Liu, H. (2023). Stylometric Detection of AI-Generated Text in Twitter Timelines. ArXiv. /abs/2303.03697