Discerning differences: A study on lexical features and vectorization techniques

Introduction

With the implementation of ChatGPT and its rising popularity, text generation and processing have taken the forefront of AI. Due to its ability to produce human-like responses, and its ability to produce other forms of media, concerns regarding its misuse have also risen. With the growth of natural language generation models (NLG), AI detection software must also stay relevant to prevent the potential threats of misuse of NLG text. Detecting and discerning between human and synthetic text is pivotal for maintaining authenticity and security in digital communication. Current AI text content detectors are becoming obsolete with each new iteration of GPT. Focus must begin shifting onto our text detection tools rather than the NLG models, as to prevent data misuse. So in this project, we tackle the text detection methods within the paper “ChatGPT Generated Text Detection” (Shijaku & Canhasi, 2023) to test the validity of their methods, and expand upon their experimentations.

TOEFL dataset

To further recreate the findings in the paper “ChatGPT Generated Text Detection” (Shijaku & Canhasi, 2023), we use the TOEFL dataset that was provided. TOEFL is a widely used standardized test to measure the English language ability of non-native speakers. The dataset contains 252 essays consisting of 126 TOEFL human essays and 126 ChatGPT essays based on the questions used in the TOEFL exams.

XGBoost

XGBoost was utilized to help recreate the experimentation conditions in the paper “ChatGPT Generated Text Detection” (Shijaku & Canhasi, 2023). XGBoost is a regularizing gradient-boosting model that utilizes decision trees. It is also a boosting ensemble learning algorithm, which means that it focuses on sequentially adding weak learners to the ensemble with a focus on correcting errors made by previous trees to minimize loss during training.

Logistic Regression

Logistic Regression was added as a baseline model to compare to XGBoost. Logistic regression utilizes a statistical algorithm to conduct classification. Logistic regression as a baseline model works well due to its simplicity and ability to process large volumes of data quickly.

Features and pre-processing

The paper uses feature extraction methods to produce a new dataset containing 244 lexical features as seen in Figure 1. The purpose of the dataset was to create more human-readable data rather than implementing tools such as TFIDF. The dataset is described below.

When this feature set was created for this experimentation, we realized that certain categories such as “frequency of character unigram (36)” did not reach its total feature count. When we reached out to the creators of the paper, it was due to a conversion error. As they accidentally based the amount of features on the Albanian alphabet rather than English. So our total lexical feature count is 239.

Figure 1: Lexical feature overview from “ChatGPT Generated Text Detection”

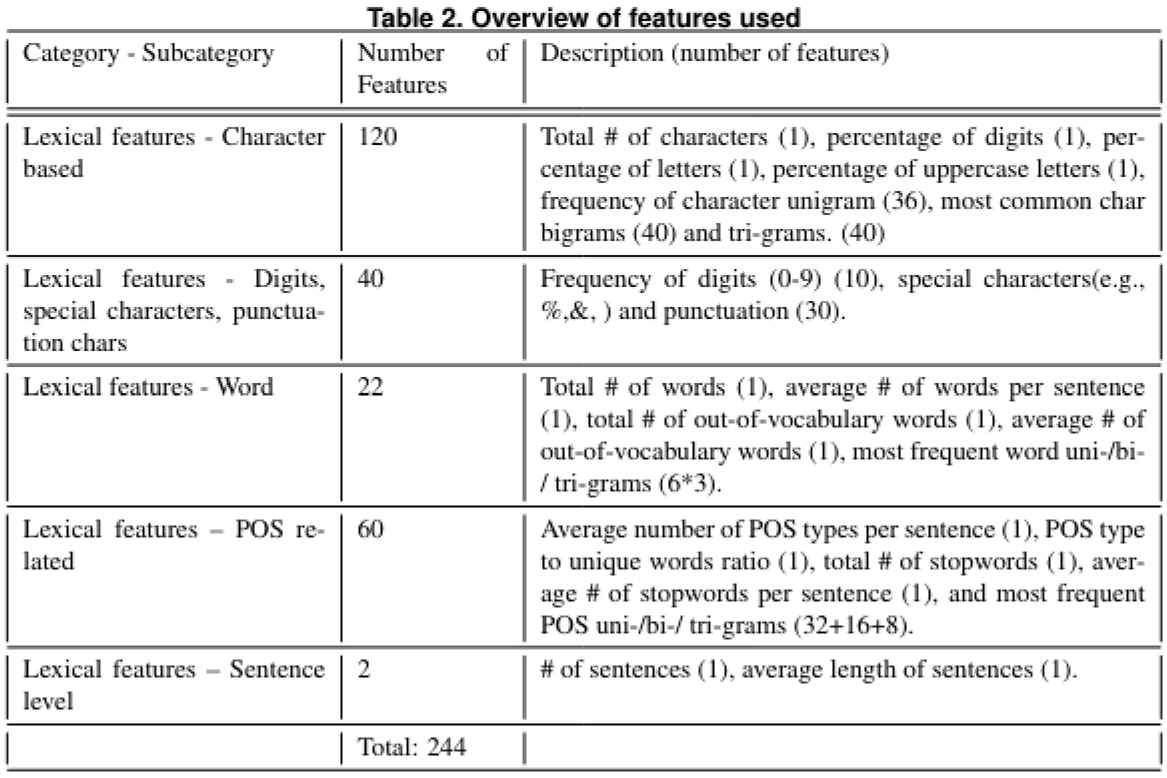


Table 2. Overview of features used

Category - Subcategory	Number of Features	Description (number of features)
Lexical features - Character based	120	Total # of characters (1), percentage of digits (1), percentage of letters (1), percentage of uppercase letters (1), frequency of character unigram (36), most common char bigrams (40) and tri-grams. (40)
Lexical features - Digits, special characters, punctuation chars	40	Frequency of digits (0-9) (10), special characters(e.g., %,&,) and punctuation (30).
Lexical features - Word	22	Total # of words (1), average # of words per sentence (1), total # of out-of-vocabulary words (1), average # of out-of-vocabulary words (1), most frequent word uni-/bi-/ tri-grams (6*3).
Lexical features – POS related	60	Average number of POS types per sentence (1), POS type to unique words ratio (1), total # of stopwords (1), average # of stopwords per sentence (1), and most frequent POS uni-/bi-/ tri-grams (32+16+8).
Lexical features – Sentence level	2	# of sentences (1), average length of sentences (1).
	Total: 244	

The paper also implements Term frequency-inverse document frequency pre-processing. TF-IDF has also been used for testing within the paper as a comparison dataset against the lexical feature dataset. TF-IDF measures word importance by calculating the frequency of which it appears. It is typically used to extract relevant word frequencies quickly for NLPs. When processed on the TOEFL essays TFIDF produces a dataset containing 5,746 features.

Approach

Before rerunning the experiments from “ChatGPT Generated Text Detection” (Shijaku & Canhasi, 2023) we run Principal Component Analysis (PCA) for the 20 most prominent features within the lexical feature dataset. This is to reduce the 239 features in the lexical feature dataset down to a key set of features. This is because we suspect that the author’s feature set may contain redundant features and features that may have high bias to the TOEFL dataset.

After running PCA we graph the variance explained by its principal component to identify the sharpest point in which variance decreases and levels off. This is known as the elbow method, which is a quick way to identify the optimal number of principal components. Using the new feature information, we filter out the non-prominent features within the lexical feature dataset to create a new PCA-filtered feature dataset.

We will then do a test train split on the new PCAfiltered feature dataset, lexical features dataset, and TFIDF dataset. Each set would be divided into a train subset that contains 80% of its data, and a test subset that contains 20% of its data.

We then recreate the experiments from “ChatGPT Generated Text Detection” (Shijaku & Canhasi, 2023) by using 5 K-fold cross-validation and evaluating XGBoost, and Logistic regression’s performance using the training subset from the PCA filtered feature dataset, lexical features dataset, and TFIDF dataset. We will be using Logistic Regression as a baseline model to compare the results of the XGBoost model. We will be obtaining results for Accuracy, Precision, Recall, and F1 score.

After comparing the scores to the original paper, we will then run GridSearchCV on XGBoost to conduct hyperparameter tuning on XGBoost. The model will be trained on the PCA-filtered feature dataset.

When we find the optimal hyperparameters for an XGBoost trained on the PCA-filtered feature dataset, we will conduct a final evaluation on XGBoost utilizing the PCA-filtered feature test subset. We will compare it to an XGBoost model that will be trained and tested on the lexical feature dataset’s subsets.

Evaluating XGBoost and Logistic Regression

Evaluation will be conducted on 3 datasets: PCA filtered feature dataset, lexical features dataset, and TFIDF dataset. Each dataset contains 252 samples and will have a train test split consisting of an 80% training subset, and 20% testing subset. The PCA filtered feature dataset after conducting PCA contains 2 features: “Total characters", and “Most frequent trigrams 5”. The lexical features dataset will contain 239 features. While the TFIDF dataset contains 5,746 features.

For our first evaluation, we will be recreating the evaluations from the original authors by using our training subsets and 5-fold cross-validation to evaluate XGBoost and Logistic regression. We will be obtaining results for Accuracy, Precision, Recall, and F1 score.

To conduct hyperparameter tuning on our XGBoost model, we will be using the PCA-filtered feature dataset's training subset. We will be using GridSearchCV and 3 CVs to find the best-contributing parameters.

XGBoost hyperparameter search space

There were many parameters that were used to optimize XGBoost. Below are the listed parameters and the range of exploration:

```
{
        'learning_rate': [0.05, 0.1, 0.2]
        'max_depth': [None, 2, 3, 5, 7, 10, 15],
        'max_leaves': [None, 2, 3, 5, 7, 10, 15],
        'subsample': [None, 0, .5],
        'colsample_bytree': [None, 0, .5],
        'colsample_bynode': [None, 0, .5],
        'colsample_bylevel': [None, 0, .5],
        'gamma': [None, 0.5, 1.0],
        'reg_lambda': [None, 2, 5.0, 10],
        'reg_alpha': [None, 2, 5.0, 10]
}
```

How hyperparameters were optimized

The hyperparameters of XGBoost were optimized using GridSearchCV. XGBoost was trained on the PCA-filtered feature dataset. After running hyperparameter tuning we are given a set of the best parameters. After testing all combinations of hyperparameters in the search space, we take the combination with the highest Accuracy.

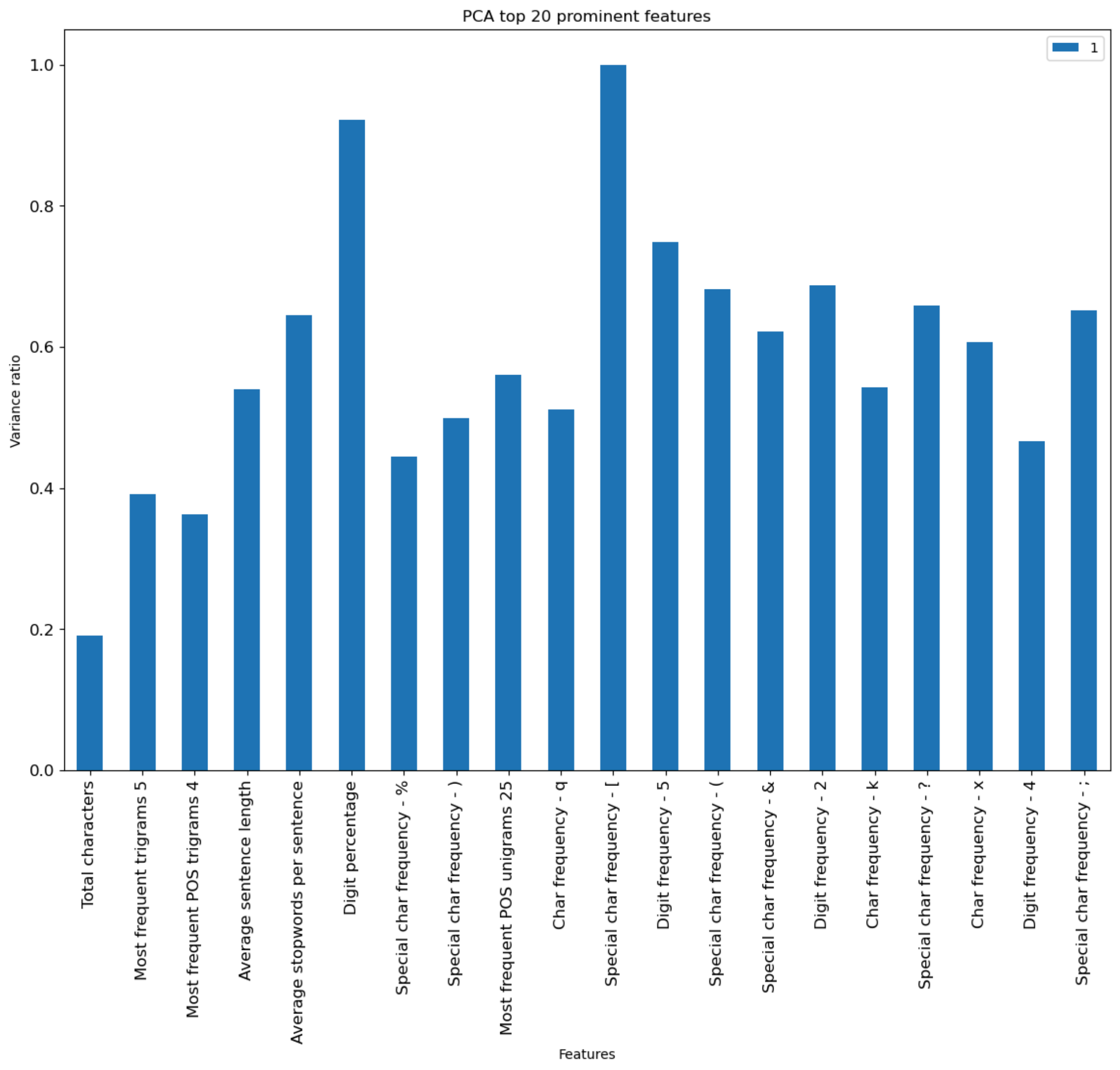
Evaluating on a clean test set

In our last evaluation, we will test our XGBoost model with the best-contributing parameters from GridSearchCV and train and test on the PCA-filtered feature dataset's subsets. We will compare it to an XGBoost model that will be trained and tested on the lexical feature dataset's subsets.

Results

Figure 2: PCA Elbow Plot and PCA bar graph

After applying PCA across 20 principal components, we plot the components by their variances and we are able to discern two outstanding features as seen in Figure 2. We then extracted the two most prominent features, “Total characters” and “Most frequent trigrams 5”, from the lexical features dataset to create a new dataset containing those two features.



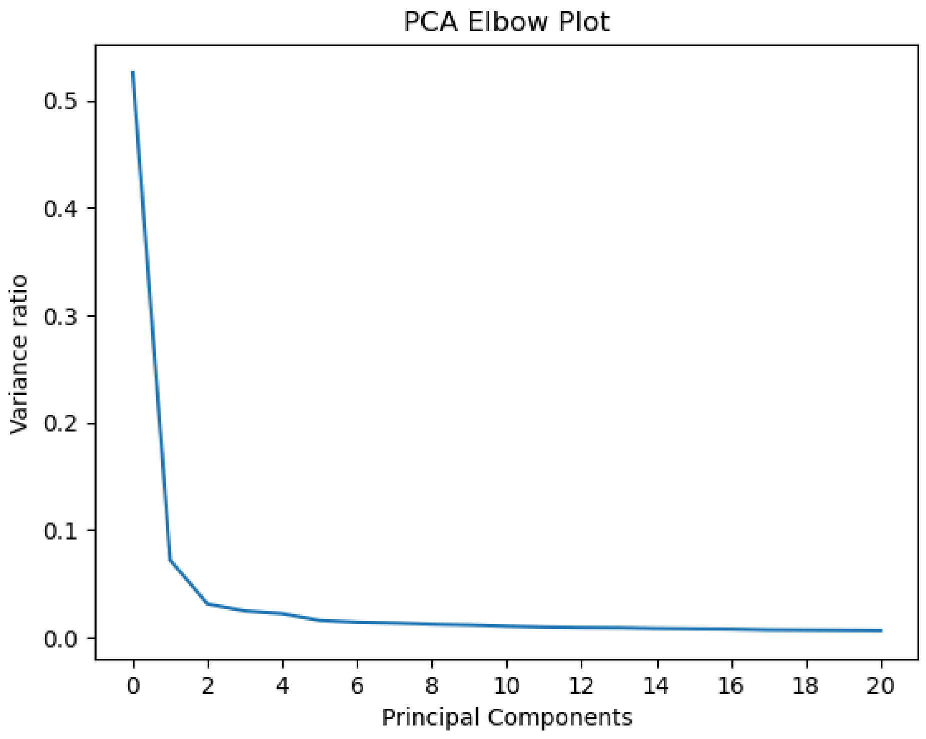


Figure 3: Experiment Recreation

We recreate the experiment process as done in the paper “ChatGPT Generated Text Detection” (Shijaku & Canhasi, 2023). Looking at Figures 2 and 4 we can see that the results were not exact. In fact, our recreation has underperformed.

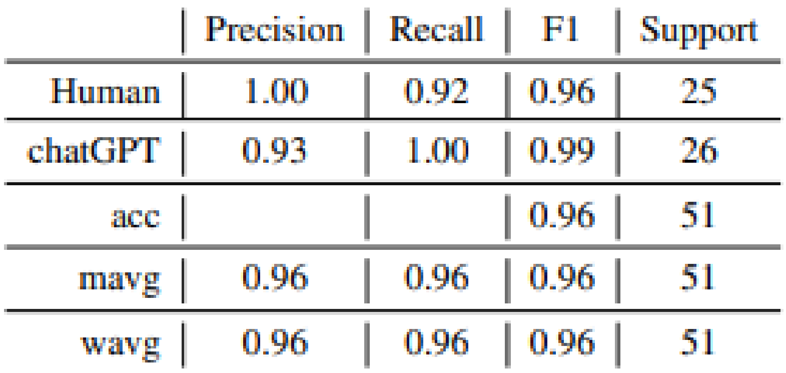
Lexical Features

Models	**Accuracy**	**Precision**	**Recall**	**F1**
XGBoost	91.55%	91.01%	93.14%	91.90%
Logistic Regression	93.05%	95.99%	90.29%	92.97%

TFIDF

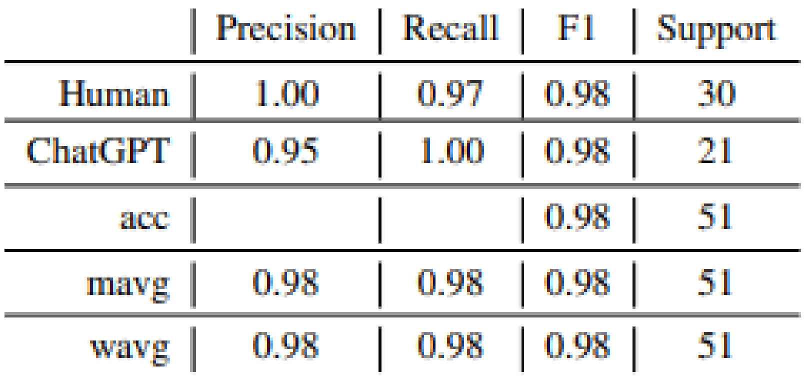
Models	**Accuracy**	**Precision**	**Recall**	**F1**
XGBoost	87.54%	92.29%	82.43%	86.92%
Logistic Regression	94.04%	97.05%	91.29%	93.89%

Fig. 4: Lexical feature results from “ChatGPT Generated Text Detection”



	Precision	Recall	F1	Support
Human	1.00	0.92	0.96	25
chatGPT	0.93	1.00	0.99	26
acc			0.96	51
mavg	0.96	0.96	0.96	51
wavg	0.96	0.96	0.96	51

Fig. 5: TF-IDF results from “ChatGPT Generated Text Detection”



	Precision	Recall	F1	Support
Human	1.00	0.97	0.98	30
ChatGPT	0.95	1.00	0.98	21
acc			0.98	51
mavg	0.98	0.98	0.98	51
wavg	0.98	0.98	0.98	51

Fig. 6: PCA Filtered Features

XGBoost trained on the PCA-filtered dataset shows a drop in accuracy from 91.55% to 75.16%.

PCA Filtered Features

Models	Accuracy	Precision	Recall	F1
XGBoost	75.16%	75.06%	77.48%	76.14%

Fig. 7: Final Evaluation

After applying the best parameters found from GridSearchCV onto XGBoost, we notice an improvement in the classification accuracy of our PCA-filtered dataset. But compared to the full features set of lexical features, we still underperform.

PCA Filtered Features

Models	Accuracy	Precision	Recall	F1

XGBoost	78.43%	87.50%	72.41%	79.25%

Lexical Features

Models	Accuracy	Precision	Recall	F1
XGBoost	90.20%	95.83%	85.19%	90.20%

Risk of Distribution Shift

XGBoost is at risk if a distribution shift occurs. This is due to the TOEFL dataset size, as the model could have overfit. When faced with new text of varying types, it would most likely fail to generalize.

Discussion

In our replication of the results from “ChatGPT Generated Text Detection” (Shijaku & Canhasi, 2023), we tested the original TOEFL dataset using TF-IDF and the custom lexical feature set. Through this we found that TF-IDF slightly underperforms compared to the Lexical feature dataset (Figure 3). Looking at the original author’s results, they have also claimed that their lexical features outperformed the TF-IDF model. Although our recreation does not match the results found in the original paper, the author’s claim is still relevant.

However, this could potentially be due to the text preprocessing conducted on our TF-IDF dataset. As the original author’s paper does not include any information for text preprocessing. We had to implement our own processes that may differ from the original experiment and therefore affect the results.

Additionally, we note that the lexical features outperform TF-IDF when running 5 K-Fold cross-validation on XGBoost. When running the same experiment on Logistic Regression we notice an increase in accuracy for TF-IDF (Figure 3). This challenges the dataset’s need for heightened complexity when a simpler model provides either comparable or outperforming outcomes.

Moving onto our final evaluation of the PCA-filtered dataset, looking at the results in Figure 7. It shows a large decrease in performance when compared to the test replication results in Figure 3. However, It is important to highlight that these scores are obtained by only using two features. This points to the model’s ability to classify features is still robust using the standout features. This proves the dominance of particular features above others. We also note in our XGBoost model training on the full lexical feature dataset (Figure 7), that the performance remains consistent to the replicated 5 K-Fold evaluation (Figure 3).

Conclusion

In conclusion, this study underscores the importance of selecting appropriate features and models for the task of detecting NLG text. While the original authors found success with lexical features and XGBoost, our findings suggest that simpler models and alternative feature extraction methods, such as TF-IDF can perform equally well or even better in certain contexts. We also note that the investigation into dimensionality reduction utilizing PCA also highlights the potential of refining the feature selection process the original authors used. Moving forward, further exploration into more advanced models and more refined feature selection processes may aid in more accurately and reliably detecting synthetic text.

Reference(s)

Dhaini, Mahdi, Wessel Poelman, and Ege Erdogan. "Detecting chatgpt: A survey of the state of detecting chatgpt-generated text." arXiv preprint arXiv:2309.07689 (2023).