Universidade de Aveiro
Departamento de Eletrónica, Comunicações e Informática

Algorithmic Information Theory (2023/24)

Lab work nº 2

08 May 2024

Rafael Pinto, nº 103379

Ricardo Antunes, nº 98275

Pompeu Costa, nº 103294

Contents

Introduction

This report addresses the implementation of a program that determines if a text is human generated or has been rewritten by ChatGPT. We used data compression algorithms to measure the similarity between texts, eliminating the need for a separate feature extraction step. For this purpose, we implemented the "was chatted" program, which calculates the estimated number of bits required to compress the text under analysis using finite-context models. These models will be trained based on the provided texts representing both texts rewritten by ChatGPT and texts written by humans. The program will build finite-context models for each class and use these models to estimate the number of bits required to compress the text under analysis. Based on this estimate, the program determines the class to which the text belongs. The report covers all stages of the work, from the development of the models to the analysis of the results obtained. Several tests will be carried out to evaluate the effectiveness of the program in classifying texts, determining whether they were rewritten by ChatGPT or not. Additionally, we will investigate the factors that may influence this effectiveness.

WasChatted

WasChatted is a program designed to determine if a given text was written by a human or ChatGPT, using pre-trained models which are FCMs (Finite Context Model), more specifically, Markov models.

WasChatted contains two main functionalities:

- Train
- Analyze

Train

The train functionality uses a text file to train the model. Upon completing the training, a binary file is created so it can be used by the user to analyse the desired text file.

There are three main parameters to train the model, as can be seen in Figure 1:

- Text file (dataset)
- Context size (k)
- Smoothing (alpha)

Although the last two parameters are optional, the value chosen for each has an impact on the results. We talk about this on the results section.

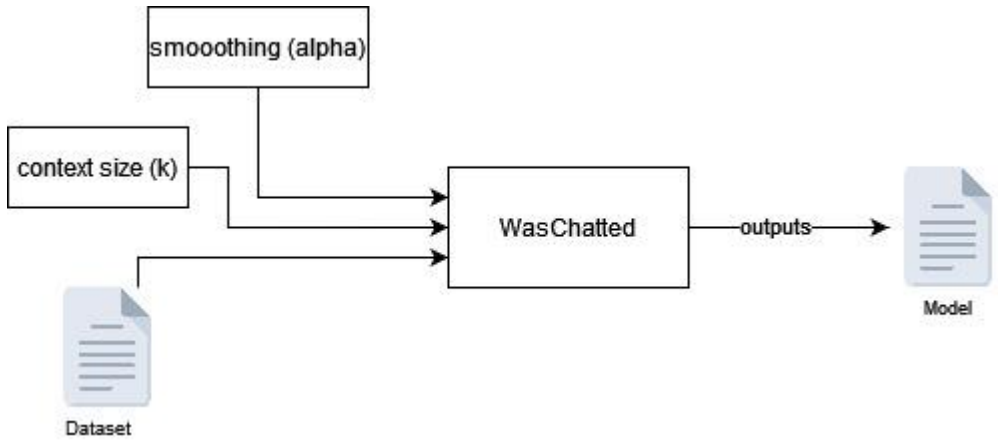

Figure 1: Train functionality of WasChatted

As mentioned previously, during training, the program uses the Markov model, which is used to calculate statistics. The Markov model saves the counting of symbols of a given context.

For example, let's consider the following text excerpt

AABBABBBBAAA

And that the desired size of the context is 2. In this case, we would get the results of Table 1.

Table 1: Example of counting table of FCM

	A	B
AA	1	1
AB	0	2
BA	1	1
BB	0	2

After training the model, the program saves the data in a binary file, as mentioned previously. This file contains the context size, the smoothing value[1] and the counting of characters for each context.

Analyze

This functionality uses the pre-trained models to determine if the provided text was written by a human or ChatGPT. This function compares the text provided with the contexts and characters stored in the pre-trained models and determines which one is the most likely. It's important to note that this process is not perfect and will make mistakes on the classification, which is noted on the results section.

There are three important parameters for this function, as seen in Figure 2:

- Text file to be analysed
- Human model
- ChatGPT model

Note that the models don't need to have been trained with the same parameters, meaning that they don't need to have been trained with the same context size and smoothing value, as shown in Figure 3.

[1] We later noted that this parameter didn't need to be specified here and that it didn't need to be saved on the binary file. It only needed to be specified in the analyse functionality, giving more flexibility to the user. However, we decided not to change because we had already generated a lot of binaries.

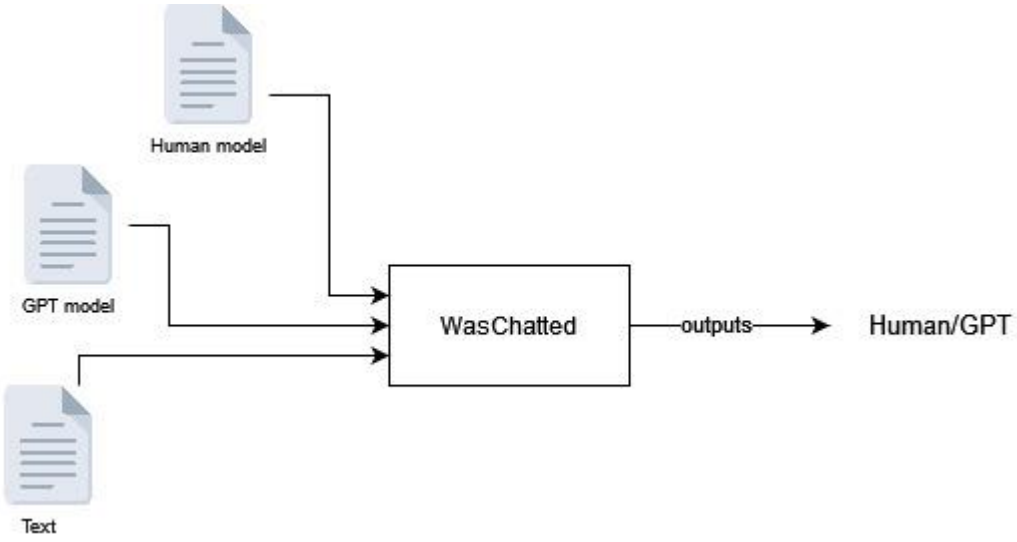

Figure 2: Analysis functionality of WasChatted.

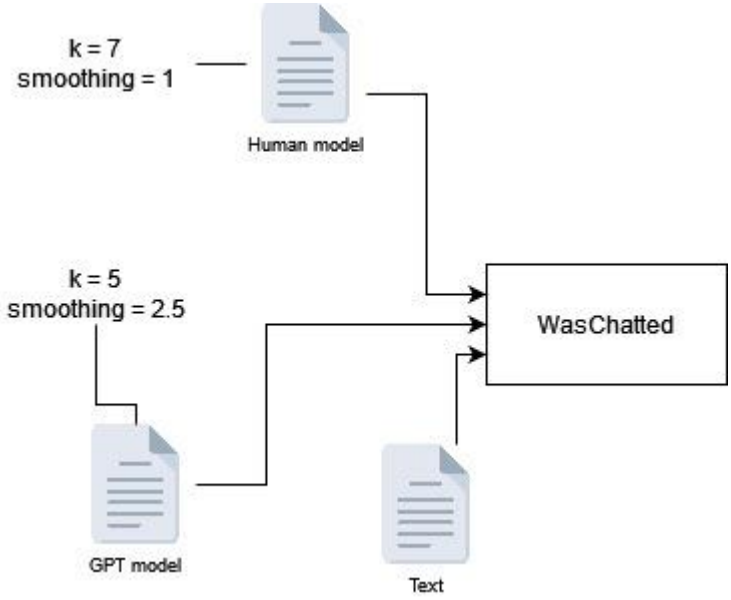

Figure 3: Example of analysis functionality with different parameters.

The analysis is done through the "compression" of a text file by both models. The result is given by the model that requires less bits after the compression, as can be seen in Figure 4.

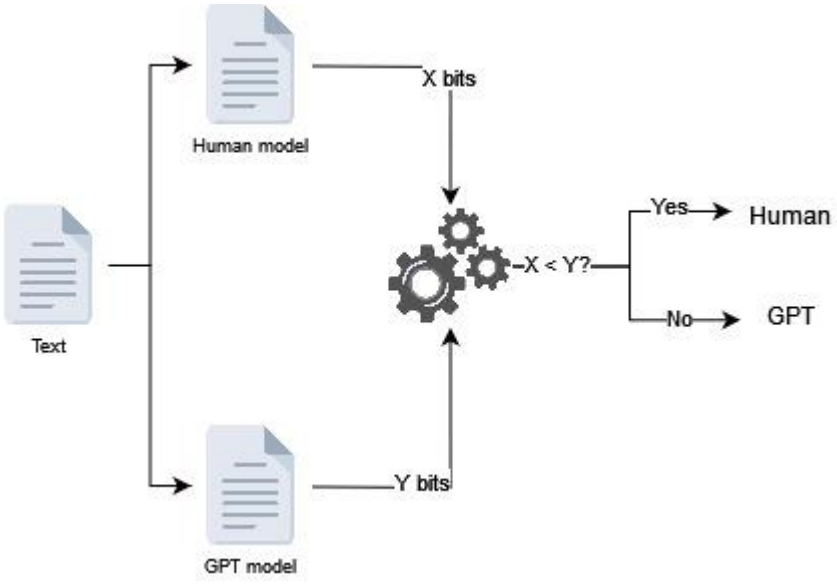

Figure 4: Classification process.

The compression is done through the probability estimation

$$P(e \mid C) = \frac{N(e \mid C)}{\sum_{s \in \Sigma} N(s \mid C)}$$

Where *e* is the character to be compressed, *C* is the context in which the character appeared, and *s* is every character that appeared in context *C*.

However, to avoid divisions by zero, the smoothing parameter is introduced

$$P(e \mid C) = \frac{N(e \mid C) + \alpha}{\sum_{s \in \Sigma} N(s \mid C) + \alpha \times |\Sigma|}$$

Where *a* is the smoothing parameter.

After calculating the probability, the program calculates the number of bits for that probability

$$B = -\log_2 P$$

Where *P* is the probability.

This is done to every character on the text file provided, which gives the total amount of bits required to represent the compressed version of the file.

This process is done by both models and the model that requires the smaller number of bits is the answer, like in Figure 4. The smaller number of bits required means that the contexts and characters of contexts on the text file, are similar to the contexts of that model.

Dataset

We used a dataset from *huggingface.co [1].* After choosing the dataset, we opted for an approach that splits the dataset into 90% of the data for training and 10% for testing. The dataset consists of human texts from Wikipedia and their corresponding rewrites generated by ChatGPT (GPT-3 model).

Results

The models with window size, K, equal to 4 were tested with 6000 texts of each class (Human or ChatGPT) of the testing set. The resulting confusion matrix, shown in Figure 5, shows that the models are more accurate predicting human texts than ChatGPT rewritten texts. Overall, these were the best models that we tested having a F1 score of 0.87.

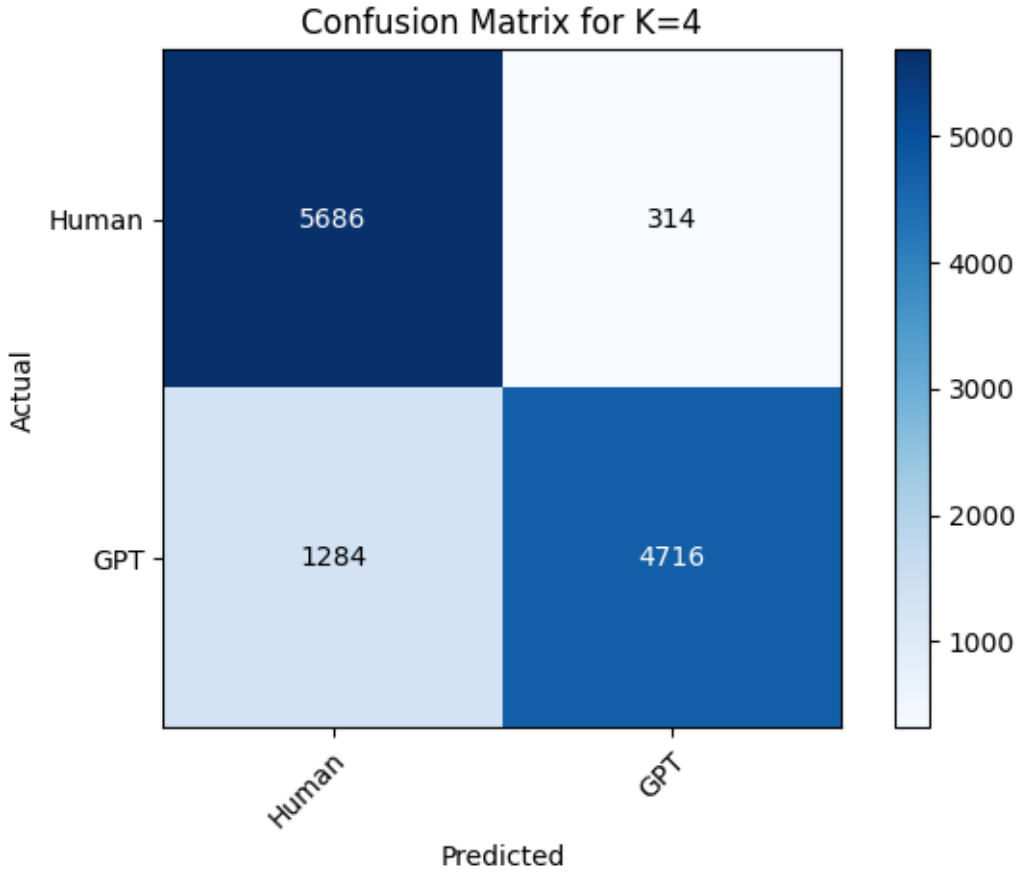

Figure 5: Confusion Matrix of the results obtained with texts from testing set

Different Configurations

It was created 5 models for each classification (5 for human generated text and 5 for ChatGPT generated text) with different values of K. Then each model was tested with the texts in the testing set to extract results for conducting an analysis to determine the optimal value of K. The F1-Score was calculated with the results of the different models. As shown in Figure 6, it is evident that the models with the highest score are those with K equal to 4.

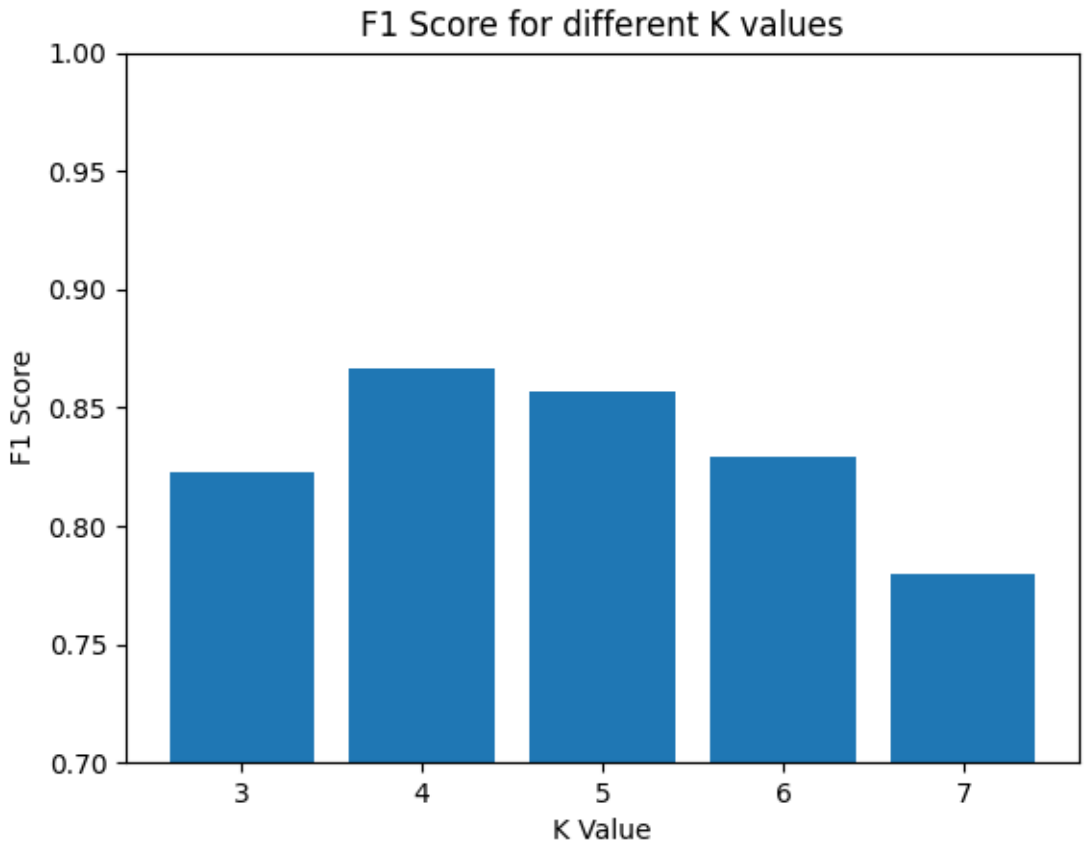

Figure 6: Bar Chart with F1 Scores of the models with different K values.

Data Outside of Dataset

The models with K equal to 4 were also tested using texts sourced from diverse sources such as articles, news websites - textual content distinct from that of the training dataset - and Wikipedia. In order to extract these examples, we collected 100 texts and asked ChatGPT to rewrite it. As expected, the models' predictions were not very accurate, resulting in an F1 score of 0.47. The confusion matrix, shown in Figure 7, shows that the models cannot predict ChatGPT rewritten text very well. We believe that this occurs, because the texts of the dataset used to train the models were rewritten by GPT-3 model and for these examples it was rewritten by GPT-3.5.

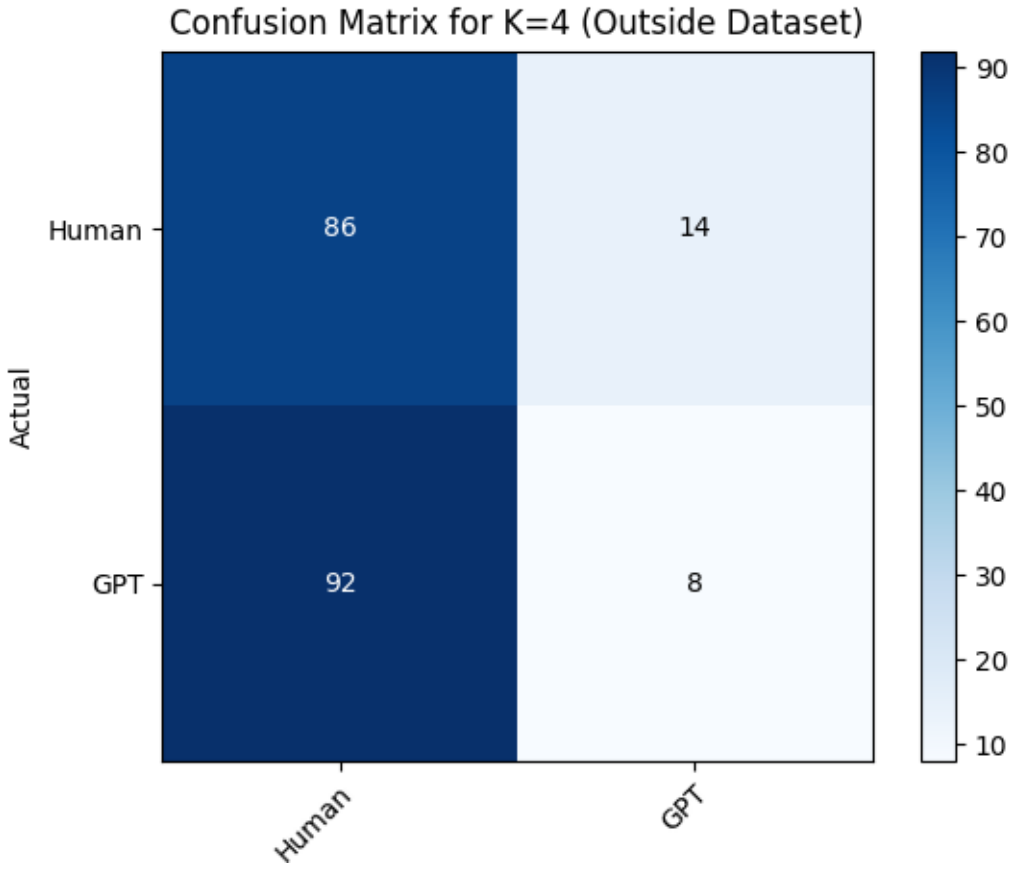

Figure 7: Confusion Matrix of the results obtained with texts outside the dataset.

Conclusion

The program developed, in this project, works as expected and implements all the features proposed, such as building finite-context models for each class and using these models to estimate the number of bits required to compress the text under analysis. Our results demonstrate that our program has relatively good performance identifying the class of a text, as shown by a F1 Score of 0.87. However, for texts with textual content distinct from that of the training dataset, it has its flaws, but we believe that is, because our ChatGPT model is trained on rewritten text from GPT3, and the rewritten text we extracted is from GPT3.5. We realized that late in the project and thus didn't have time to find another dataset to train our ChatGPT model.

References

[1] aadityaubhat/GPT-wiki-intro · Datasets at Hugging Face. (n.d.)

https://huggingface.co/datasets/aadityaubhat/GPT-wiki-intro