AI Text Detector

Alexandre Gazur
Departamento de Eletrónica,
Telecomunicações e Informática
Universidade de Aveiro
Aveiro, Portugal
alexandre.gazur@ua.pt

Daniel Ferreira
Departamento de Eletrónica,
Telecomunicações e Informática
Universidade de Aveiro
Aveiro, Portugal
djbf@ua.pt

Ricardo Pinto
Departamento de Eletrónica,
Telecomunicações e Informática
Universidade de Aveiro
Aveiro, Portugal
rjsp15@ua.pt

João Matos
Departamento de Eletrónica,
Telecomunicações e Informática
Universidade de Aveiro
Aveiro, Portugal
joaorpm02@ua.pt

***Abstract*—The rapid advancement of powerful artificial intelligence technology has brought forth various challenges, including the emergence of AI-generated text that often blurs the line between human-authored and machine-generated content. In this project, we propose a browser extension aimed at detecting AI-generated text in web pages and PDFs. The extension integrates with Model Hub, a web-based platform, allowing users to submit their AI text detection models and use them within the extension. This project contributes to AI transparency and enables informed decision-making in an era of increasing AI-generated information.**

***Index Terms*—AI Text Detection, ChatGPT, OpenAI, Web extension, Web application, RESTful API, RESTful API, Artificial Intelligence, Machine Learning, Natural Language Processing, Language Models**

I. Introduction

The widespread availability of AI language models, such as OpenAI's GPT-3, has democratised the creation of text that closely resembles human-generated content. As a result, distinguishing between AI-generated and human-authored text has become increasingly complex [1]. While AI-generated text offers benefits in various applications, including content creation and language translation, it also raises concerns about the potential for misinformation, propaganda, and manipulated narratives.

Our motivation stems from the need to tackle the complexities of AI-generated text. Users often question whether the content they encounter is written by a human or an AI, leading to trust issues and hindering information consumption. We aim to alleviate this uncertainty and enable users to make informed decisions about the content they engage with.

The primary goal of our project is to develop a browser extension capable of detecting AI-generated text in web pages and PDF documents. The extension provides users with a straightforward and reliable tool to identify and distinguish between human-authored and AI-generated content. The UI is designed to be comprehensive and intuitive, providing customisation options to cater to individual preferences.

Additionally, our project aims to embrace the collaborative nature of AI development. By allowing users to submit their own AI text detection models through our Model Hub web platform, we ensure that the system maintains itself, stays up-to-date with the latest natural language processing techniques and machine learning algorithms, and remains helpful and effective as new models and approaches emerge.

Through these efforts, we aim to contribute to the promotion of transparency, trust, and critical thinking in the digital era, while also facilitating continuous improvement and innovation in the detection of AI-generated text.

II. Software architecture

The technological model in Figure 1 gives an overview of the technologies used by the system and how they interact with each other.

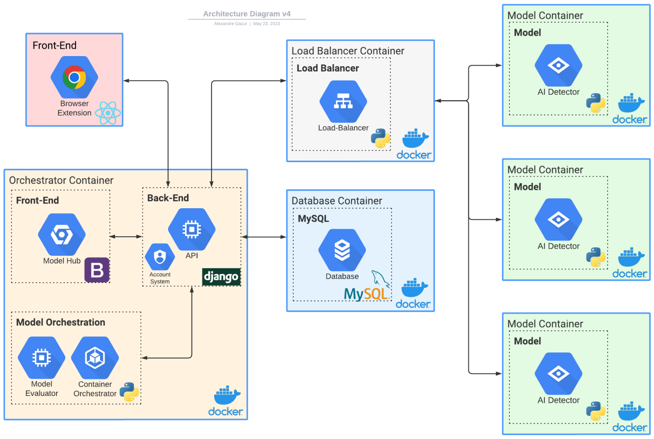
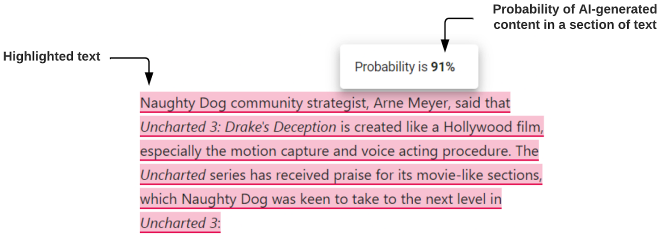


Fig. 1. Software architecture

The system consists of a Plasmo and React web extension that communicates with a Django Backend and API. Users can request text analysis with different models. The Backend serves as the intermediary, facilitating communication between the extension and the models. There are two types of models: API-based and script-based. For API models, the Backend retrieves the relevant API URL from the database and sends a request to analyse the text, waiting for the probability result. Script models are handled differently, as the Backend communicates with the Load Balancer, which forwards the request to the appropriate container running the chosen model. Additionally, the Model Hub website allows users to submit models, which undergo evaluation by the Model Evaluator against a dataset of classified text. Accepted models are stored in the database, and for script models, the Container Orchestrator initiates a Docker container for their execution.

III. Implementation Details

The extension highlights detected AI-generated text using customisable colours, illustrated in Figure 2. This visual cue enables users to quickly identify and differentiate between AI-generated and human-written content.

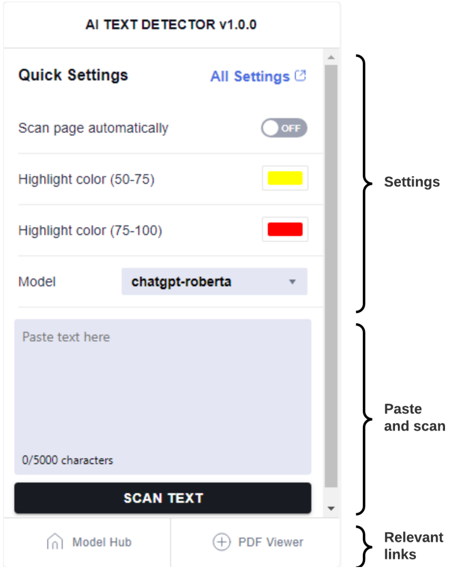


Probability is **91%**

Naughty Dog community strategist, Arne Meyer, said that *Uncharted 3: Drake's Deception* is created like a Hollywood film, especially the motion capture and voice acting procedure. The *Uncharted* series has received praise for its movie-like sections, which Naughty Dog was keen to take to the next level in *Uncharted 3*.

Fig. 2. Example of highlighted text and score

The settings of the extension, Figure 3, are conveniently displayed within the popup interface, allowing users to customise and fine-tune the extension according to their individual preferences.



AI TEXT DETECTOR v1.0.0

Quick Settings

Scan page automatically: OFF

Highlight color (50-75): 🟨

Highlight color (75-100): 🟥

Model: **chatgpt-roberta**

Paste text here

0/5000 characters

Fig. 3. Popup of the extension

IV. Evaluation and Testing

The usability of the system was evaluated using the System Usability Scale [2]. The study included 8 participants, primarily students from the LEI course at the University of Aveiro. The results of the study are shown in Table I. The final SUS score we obtained is 92.2 and represents the average of all respondents. This indicates that the respondents had a positive outlook on the use of the application.

TABLE I
System Usability Scale Scores

Statement	1	2	3	4	5
I think that I would like to use this system frequently	0	3	2	1	2
I found the system unnecessarily complex	8	0	0	0	0
I thought the system was easy to use	0	0	0	0	8
I think that I would need the support of a technical person to be able to use this system	7	1	0	0	0
I found the various functions in the system were well integrated	0	0	0	2	6
I thought there was too much inconsistency in this system	4	4	0	0	0
I would imagine that most people would learn to use the system very quickly.	0	0	0	0	8
I found the system very cumbersome to use	8	0	0	0	0
I felt very confident using the system	0	0	0	2	6
I needed to learn a lot of things before I could get going with this system	6	2	0	0	0

We also gathered feedback on the strengths and weaknesses of the application through a narrative approach. Some of those suggestions were later adopted in subsequent iterations.

V. Conclusion

The project has achieved significant milestones, resulting in the successful development of an application that surpassed the features and value of existing solutions. The main goals have been accomplished, and the final version incorporates a flexible architecture for easy integration of additional models and future use cases. The system demonstrated high usability during testing, though limitations in validity may exist due to potential biases in the respondent pool. Overall, the project has been a success, offering a valuable browser extension with the potential for expansion and adaptation to emerging technologies in the NLP and ML fields.

References

[1] Chakraborty, Souradip and Bedi, Amrit and Zhu, Sicheng and An, Bang and Manocha, Dinesh and Huang, Furong. (2023). On the Possibilities of AI-Generated Text Detection.

[2] Brooke, John. (1995). SUS: A quick and dirty usability scale. Usability Eval. Ind.. 189.