Engineering Educators Bringing the World Together
2025 ASEE Annual Conference & Exposition
Palais des congrès de Montréal, Montréal, QC • June 22–25, 2025
Paper ID #46338

Assessment of Large Language Models for Wildcard Match Identification

Claire Han
Abel Andres Reyes-Angulo, Michigan Technological University

ABEL REYES-ANGULO is a Ph.D. student in Computational Science and Engineering at Michigan Technological University in Houghton, MI. He received his B.S. degree in Computer Engineering and his M.S. degrees in Electrical and Computer Engineering from Purdue University Northwest in Hammond, IN, in 2018 and 2021, respectively. His research interests include, but are not limited to, machine learning, deep learning in computer vision, medical image analysis with AI, software development, and virtual reality.

Dr. Sidike Paheding, Fairfield University

Assistant Professor in the Department of Computer Science and Engineering.

Assessment of Large Language Models for Wildcard Match Identification

January 2025

Abstract

Throughout its development, the internet has become a massive part of everyday life. Convenience in communication, entertainment, exercise, and various other areas have provided positive impacts for people around the world. However, with these milestones, the internet has simultaneously become a hotbed for malicious actors. These malicious actors continuously compromise private information and software systems through phishing emails, hacking, and WiFi networks. As these attackers evolve their tactics, cybersecurity is becoming a key field in computer science and must advance alongside new problems. This paper investigates the impact that Artificial Intelligence (AI) has on web browser cookies and wildcard features, which are special symbols that can represent other characters and sort files. Since wildcards organize cookies into groups that have similar data and function trends, browsers can determine which cookies are problematic earlier on. AI is an expert on observing and analyzing trends in data; including AI in the wildcard features would provide an even faster solution for web browsers to see which cookies are secure. Currently, there is plenty of research focused on cybersecurity, including the concerns of fingerprinting, phishing, and cross-site attacks, but web browsers, and the cookies in those browsers, are an overshadowed topic. Since cookies often provide security and user convenience in various websites, they are critical to ensure search history and advertising privacy for everyday users. On the contrary, if cookies are not well-researched or protected, malicious actors can take advantage of vulnerable marketing, analytics, and third-party cookies to steal personal information and expose online profiles. Furthermore, in today's society, AI is one of the most useful tools in cybersecurity. Though developments in AI are still in progress, it proves to be extremely important in learning new hacking techniques, defending systems against attacks, and refining algorithms for machines and security software. Even in the world of browsers, AI is beneficial, as it can observe patterns in dangerous cookies to preserve user privacy, extract cookie features to analyze individually, and examine wildcards to expedite data requests and organization. This research assessed web browser security through the use of Large Language Models (LLMs), such as GPT2, T5, and Flan-T5, as AI algorithms for identifying wildcard features in cookies. By inputting a long sequence of information about a cookie, such as stating its domain, function, and retention period, the LLM responds back with a yes or no answer about whether the cookie is a wildcard match. The algorithm was trained on a diverse dataset from the Open Cookie Dataset, showing a potential breakthrough for cybersecurity and AI to secure online safety. Preliminary results provide a promising performance of above ***95%*** accuracy and ***0.77*** MCC score in cookie wildcard match identification.

***Keywords*—** Cybersecurity, cookies wildcards match, Large Language Models

1 Introduction

In today's society, everyday life is heavily dependent on technology. The world has become more interwoven, and communication, entertainment, and knowledge have become convenient, leading to the widespread use of electronics and the Internet. Now, from watching television shows to checking bank finances, the Internet has become a significant part of society. However, this rapid technological progress comes with both benefits and harms.

Although the Internet is used worldwide for global communication and productivity in the workforce, alongside numerous other benefits, this web has become a massive target for cyberattacks. Malicious actors continuously seek to steal personal or sensitive information, weaken security systems, manipulate data, and exploit vulnerabilities in software. Even worse, these threats change at fast paces, creating constant new dangers that must be explored and addressed quickly.

Due to these problems, cybersecurity has become a topic of increasing importance in recent years, and there has been much research regarding common crimes, such as Malware, Denial of Service, Phishing, and AI-related attacks[1]. For instance, by training employees properly, consistently updating technological systems to be more efficient and secure, and downloading antivirus software, Malware attacks can be largely prevented[2].

Despite these massive improvements, research supporting the booming cybersecurity field is still expanding adapt to changes and become more widespread. In recent years especially, browser security has been a concerning topic, leading to more research in the area. Browsers are used in everyday life, from simple Google searches to the storage of important files in cloud drives, but have also become mired with attacks. Malicious actors attempt to steal personal information to impersonate people, leak sensitive information from drives, and damage software systems in computers through viruses and dangerous links on browsers. In response to these developments, including fingerprinting and cross-site scripting attacks, privacy-focused browsers, private extensions, and cookie setting options have appeared to give users more choices[3].

Specifically regarding cookies, third-party, advertising, and malicious cookies pose significant risks, as they can collect private information and transmit it to corporations or malicious actors, often without the user's consent or awareness[4]. These issues demand a rise in privacy; eliminating harmful cookies is crucial to the safety of citizens, and one solution is to examine cookie wildcards. Wildcards are special symbols present only in certain cookies and can represent characters found in other cookies. These symbols can simultaneously sort cookies into groups by function and type, a key to identifying malicious cookies early on.

Though browsers can skim through all possible cookies from a website and sort them manually, AI is now on the rise. Given its huge potential, AI makes processing, observing, and sorting much easier. Large Language Models (LLMs) that utilize machine learning to improve performance metrics can quickly sort cookies into the proper groups and alert browsers of any possible dangers, speeding up the process. This paper aims to compare LLMs in their performance metrics to determine which model would be most helpful to web browser security today.

2 Related Work

The emergence of complex web ecosystems has elevated cookies as essential components for enhancing web browsing functionality, user experience, and personalized content delivery. However, as cookies serve to store user data and track online activities, they have also become prime targets for malicious exploitation, raising privacy and security concerns among internet users. Multiple studies have analyzed the impacts of cookies on user privacy and explored defensive mechanisms to mitigate risks associated with third-party tracking and cookie-based fingerprinting. For instance, Englehardt and Narayanan[5] explored tracking methods that include cookies and fingerprinting across major websites, revealing a wide array of tracking techniques that compromise user privacy. They highlighted how cookies could be leveraged for persistent tracking and underscored the need for sophisticated detection and blocking techniques.

In the context of cybersecurity, recent advances in artificial intelligence (AI) have shown promise in identifying patterns within data that might indicate security risks, including malicious cookies. One such approach includes the use of Large Language Models (LLMs) like GPT2[6] and T5[7] for analyzing cookie metadata to identify potentially harmful cookies by matching wildcard patterns. Wildcards are used in cookies to generalize groups of cookies with similar data attributes, potentially highlighting patterns in cookies that may reflect security or tracking behaviors across domains. Studies such as that by Acar et al.[8] demonstrated that many cookies adopt predictable structures, and this predictability could be used to identify suspicious cookies before they compromise user privacy.

LLMs have proven particularly effective in tasks that require pattern recognition and classification based on textual information, as demonstrated by Brown et al.[9] with GPT-3 and by Raffel et al. with T5. These models, trained on extensive corpora, excel at understanding natural language patterns, making them suitable candidates for classifying cookies by type or identifying wildcard matches based on metadata descriptors. Prior research has shown that LLMs are capable of zero-shot and fine-tuned performance, which enables their deployment in tasks with minimal training data, such as identifying novel cookies based on existing patterns[6].

By applying these models to cookie identification tasks, this research attempts to leverage the high accuracy of LLMs in handling wildcards and enhance web security through early identification of potentially harmful cookies.

Given the current cybersecurity landscape, numerous studies have documented the effectiveness of machine learning and AI in security applications, from malware detection[10] to phishing identification[11]. Large language models, due to their capability to interpret complex text structures and learn from limited examples, present an advantageous approach for addressing cookie security concerns.

This research builds on previous work in exploring browser security tools that incorporate AI for real-time monitoring and classification of cookies. Prior approaches predominantly focused on detecting tracking through fingerprinting techniques or implementing static defenses against cookies with predefined characteristics. Studies such as that by Sommer and Paxson[12] have demonstrated the potential of machine learning for network intrusion detection, which parallels the approach of applying AI models to identify potentially harmful patterns within browser cookies. This study advances the field by employing LLMs to dynamically assess cookies based on wildcard patterns.

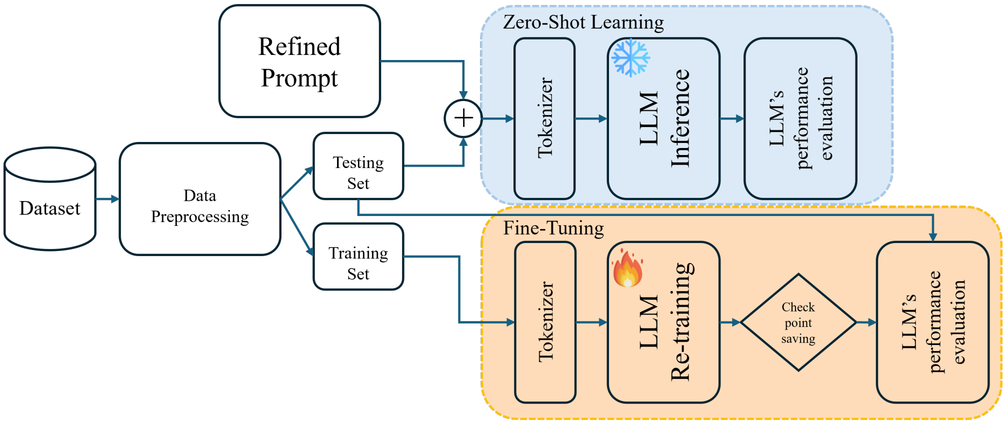


Figure 1: Framework for assessing the performance of Large Language Models (LLMs) in wildcard match classification. The workflow consists of two approaches: zero-shot learning and fine-tuning. The dataset is tokenized and passed to LLMs, followed by performance evaluation. For fine-tuning, the process includes re-training with checkpoint saving for optimization.

3 Methodology

We adopted a systematic approach to evaluate and compare the performance of various LLMs in identifying wildcard-match cookies. Figure 1 outlines the experimental framework, composed of two approaches—zero-shot learning and fine-tuning.

3.1 Dataset

In order to proceed with the comparison and experimentation of LLMs, there needed to be a diverse dataset to test from. The Open-Cookie-Database[1] is a large dataset, in which users can freely add different types of cookies and their features, such as "Platform," "Category," "Domain," "Data Controller," and "Wildcard Match." This dataset aims to organize and identify all of the major types of cookies, ultimately raising awareness on cookie security in various websites.

The next step was to prepare the dataset for use. The data started off in tables in a CSV file, but the LLMs could not process or analyze the data in that form. Thus, the goal was to prepare the CSV file in Google Colab, the only platform in these comparisons, to preprocess the data and make it available to all of the Machine Learning models. The CSV file was uploaded to Google Colab and was read and stored in tables in Python. Afterwards, all of the categories except for "Wildcard Match" were combined into one long sentence of information. This column would later be the input to the LLMs so that each model could predict whether the cookie was a wildcard. The proceeding step was to split the dataset into training and testing subsets. 80% of the data was randomly categorized in training, while the remaining 20% was given to the testing subset. The experimentation started once the data was split; each notebook had the same preprocessing steps, and each one contained a different LLM model to predict the training data.

Despite containing 2,182 cookies in total, the dataset exhibited a significant class imbalance, as shown in Figure 2: only around 243 cookies contained wildcard matches, while the majority (1939) did not. This imbalance increased the challenge of accurately

[1] Open-Cookie-Database: https://github.com/jkwakman/Open-Cookie-Database

identifying wildcard cookies. To mitigate this issue, the random splitting was done in a stratified manner, preserving the same ratio of wildcard to non-wildcard cookies in both training and testing subsets. This approach allowed the LLMs to receive a more realistic and representative sample of cookie types during training, thereby improving the likelihood of correct classification in the testing phase.

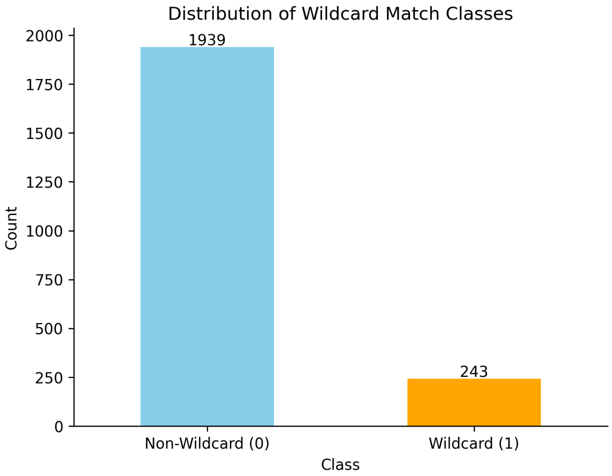


Figure 2: Class distribution of the Open-Cookie-Database. The sampling frequency, where ‘1’ represents a wildcard match, reveals a heavily unbalanced class distribution.

3.2 Evaluation Metrics

In order to analyze and compare the LLMs, a variety of evaluation metrics were employed to provide a more objective and comprehensive view of model performance. The first metric was **accuracy**, which measures how often the LLM predicts the correct category for a given input. In our setup, the input consisted of all information gathered for a cookie (e.g., Domain, Name, and Platform), and the output indicated whether or not the cookie was a wildcard match. Formally, accuracy captures the proportion of correct classifications and is mathematically expressed as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

where TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively. In this context, TP represents a correct identification of a wildcard match, TN represents a correct identification that the cookie is not a wildcard match, FP occurs when a cookie is incorrectly classified as a wildcard match, and FN occurs when a wildcard match is missed.

Precision quantifies the correctness of positive (wildcard) predictions by focusing on how many predicted positives are actually positive:

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{2}$$

Recall measures how well the model detects all actual positives:

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{3}$$

F1 score, the harmonic mean of precision and recall, provides a more robust view in the presence of class imbalance (only a small fraction of cookies are wildcard matches):

$$F1 = \frac{2 \times TP}{2 \times TP + FP + FN} = 2 \times \frac{P \times R}{P + R}, \tag{4}$$

where P and R denote precision and recall, respectively.

Finally, the **Matthews Correlation Coefficient (MCC)** provides a more balanced view of binary classification by incorporating all four confusion matrix components. Unlike accuracy, MCC remains robust even when the dataset is heavily skewed toward one class, making it particularly recommended for problems in the Open-Cookie-Database. Its value ranges from -1 (total disagreement) to 1 (perfect agreement):

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}. \tag{5}$$

By considering the full confusion matrix, MCC offers a fairer comparison of model results for imbalanced datasets[13].

3.3 Large Language Models (LLMs)

Large Language Models (LLMs) are advanced neural network architectures designed to understand and generate human-like text by learning patterns from vast amounts of textual data. Utilizing transformer architectures[14], LLMs capture long-range dependencies and contextual relationships in language, enabling them to perform a wide array of natural language processing tasks such as translation, summarization, and text classification. Their ability to generate coherent and contextually relevant text makes them valuable tools in applications like cybersecurity, where they can analyze and interpret complex textual data for threat detection and analysis[9].

In this work, there is an assessment of the performance of openly available LLMs (as shown in Table 1) that are suitable for limited computational resources in identifying wildcard-match cookies. Through zero-shot and fine-tuning experiments, many observations and evaluations are made.

3.3.1 GPT2

GPT2, developed by OpenAI, is a transformer-based language model renowned for its capability to generate coherent and contextually appropriate text[6]. Trained on a diverse dataset of internet text, GPT2 can perform various language tasks without task-specific training data, making it versatile for applications requiring text generation and understanding. Its architecture allows it to predict subsequent words in a sequence, enabling it to produce human-like text that can be leveraged for tasks such as content creation, summarization, and, in this context, analyzing cookie metadata for security purposes.

3.3.2 T5

The Text-to-Text Transfer Transformer (T5) model, introduced by Google, approaches all-natural language processing tasks in a unified text-to-text framework[7]. In other words, both the input and output are always text strings, allowing T5 to be fine-tuned on a wide range of tasks using the same model architecture and training objective. T5 has demonstrated strong performance across multiple NLP benchmarks by converting various tasks into a text generation problem, making it a flexible choice for applications like cookie classification and wildcard matching in cybersecurity.

3.3.3 Flan-T5

Flan-T5 is an enhanced version of the T5 model that incorporates instruction fine-tuning[15]. By training on a mixture of tasks phrased as instructions, Flan-T5 improves its ability to follow task descriptions and generalize to new tasks. This makes Flan-T5 particularly effective in zero-shot and few-shot learning scenarios, where the model needs to perform well on tasks it has not explicitly been trained on. In the context of identifying wildcard matches in cookies, Flan-T5's improved understanding of instructions can lead to more accurate and reliable classification results.

4 Results

4.1 Experimental Setup

The experimental workflow comprised four main steps: selecting a relevant dataset, preprocessing the data, testing various Large Language Models (LLMs) using zero-shot learning, and fine-tuning select models for performance improvement.

[2] GPT2 Small on Hugging Face: `https://huggingface.co/gpt2`

[3] T5 Small on Hugging Face: `https://huggingface.co/t5-small`

[4] Flan-T5 Small on Hugging Face: `https://huggingface.co/google/flan-t5-small`

Model	Year of Release	# of Parameters
GPT2-Small[1]	2019	117 million
GPT2-Medium	2019	345 million
GPT2-Large	2019	762 million
GPT2-XL	2019	1.5 billion
T5-Small[2]	2019	60 million
T5-Base	2019	220 million
T5-Large	2019	770 million
T5-3B	2019	3 billion
T5-11B	2019	11 billion
Flan-T5 Small[3]	2022	80 million
Flan-T5 Base	2022	250 million
Flan-T5 Large	2022	780 million
Flan-T5 XL	2022	3 billion
Flan-T5 XXL	2022	11 billion

Table 1: Comparison of the LLM used in this study: GPT2, T5, and Flan-T5 models.

4.1.1 Zero-Shot Learning

In order to use these LLMs for wildcard match identification, the experiments started off by testing each model through zero-shot experiments. A zero-shot experiment, or zero-shot learning[16,17,18], is when a model is tested on datasets that it has never seen before during its training. As a result, the model must adapt to interpret the prompt well and output a valid answer. In these tests specifically, the entire cookie dataset was split into a trial set and a testing set, and only the testing set was considered for the performance evaluation.

The prompts inputted into the models started off basic, as shown in Table 2, but this strategy proved to be a limitation in these experiments. For example, the prompt used for T5-base was “Reply yes or no if a wildcard match: ” with all of the cookie’s gathered information added afterward. T5-base was a fairly sensitive model and responded well to a relatively vague prompt. However, upon testing T5-small, a model that has a simpler architecture and less training, the same prompt resulted in strange answers, such as “: CookieYes” or “if.” Therefore, to produce legible answers, the prompt was changed to “Does this information correspond to a cookie wildcard match, only true or false as answer: ” and the cookie’s information was added on afterwards. Alongside this change, the number of beams, in the beam search algorithm, was increased to generate better responses, improve output quality, and slow down processing time. T5-small recovered quickly and answered well with these two changes. The later models, such as the GPT2 models, T5-large, and Flan-T5-models, all required a change in prompts, as well, because they also output nonsensical answers, including “Yes No” and “If you are using a third party service.” Although the number of beams stayed consistent after the T5-small model, the prompts varied, as each LLM responded to different wording types. These limitations did produce challenges in gathering the proper data but were overcome in the later stages.

A vast majority of the models ran smoothly, yielding a large collection of data in response to cookie wildcard matches. The experiments were conducted on the free version of Google Colab, utilizing the GPU runtime (NVIDIA T4 GPU). While this setup handled most models effectively, some advanced models, such as T5-large and GPT2-XL, required significant computational resources and storage, leading to frequent crashes during sessions.

To address these challenges, these models were allocated individual Google Colab sessions, optimizing the available resources by clearing storage and restarting sessions as needed. These cautionary measures ensured that even the larger models with more complex architectures could run under similar experimental conditions. However, Flan-T5-XXL posed a persistent issue—it consistently crashed due to the limited GPU memory available in the free-tier environment. Despite these obstacles, the adaptation to Google Colab’s resource constraints was manageable for the majority of models, with only Flan-T5-XXL remaining non-functional under problematic conditions.

LLM model	Prompt
T5-base	"Reply yes or no if a wildcard match: " + *text*
T5-small	"Does this information correspond to a cookie wildcard match, only true or false as answer: " + *text*
T5-large GPT2 GPT2-Medium GPT2-Large GPT2-XL Flan-T5-Large Flan-T5-XL	"Answer 'Yes' or 'No': Does the following information correspond to a cookie wildcard match? " + *text*

Table 2: Prompts used for different LLM models while performing the zero-shot learning experiments."***text***" refers to the combination of all the input features information on each sample (e.g., Domain, Name, and Platform)

Table 3: Performance Metrics for Various Models through zero-shot learning. The best results are in **bold**. *Ran out of storage in GPU

Model	Accuracy	Recall	Precision	F1 Score	MCC
T5-base	0.6270	0.6443	0.9091	0.7541	0.0000
T5-small	0.7300	0.8067	0.8792	0.8414	-0.0575
T5-large	0.8810	0.9923	0.8871	0.9367	-0.0295
Flan-T5-Small	**0.8879**	**1.0000**	0.8879	**0.9406**	0.0000
Flan-T5-base	0.1258	0.0155	1.0000	0.0304	0.0419
Flan-T5-XL	0.8375	0.9407	0.8838	0.9114	-0.0538
Flan-T5-XXL	_*Resource limitations_				
GPT2	**0.8879**	**1.0000**	0.8879	**0.9406**	0.0000
GPT2-medium	**0.8879**	**1.0000**	0.8879	**0.9406**	0.0000
GPT2-large	0.8490	0.8870	**0.9510**	0.9179	-0.0120
GPT2-XL	**0.8879**	**1.0000**	0.8879	**0.9406**	0.0000

4.1.2 Fine-tuning LLM

Through the zero-shot experiments, it became clear that the models were not well-trained in identifying cookie wildcard matches. The zero-shot experiments were the models' first exposure to cookie wildcards; in order to improve their performance metrics and help the models learn even more, the testing dataset was used to fine-tune each model and improve the models' experiences. Fine-tuning, in particular, is a machine learning strategy in which a trained model is tested on a smaller dataset to better its performance in a specific area[19,20]. Each LLM was already trained from the trial dataset and zero-shot experiments, but the testing dataset and fine-tuning experiments targeted model efficiency and quality performance in identifying whether a cookie was a wildcard match.

Fine-tuning proved to be more difficult to execute than the zero-shot experiments, as it took much longer and used up more memory than the original testing. The prompts given to each LLM were simply the data gathered about individual cookies, and every LLM was given the same prompt during fine-tuning. This process involved multiple rounds of training for different numbers of epochs (5, 10, 15, 20, and 25), allowing models to iteratively learn from the data. The models that were more basic and had simpler architecture, such as T5-small, GPT2, and Flan-T5-small, were able to run smoothly through the fine-tuning experiments. However, more complex and developed models, including Flan-T5-XL and T5-large, kept crashing and could not be modified. Errors about memory, storage, and length of time continued appearing during these experiments due to the technology and learning advancements.

Attempts to address the resource limitations included reducing batch sizes and clearing GPU memory, strategies that were also employed during the zero-shot experiments. Despite these efforts, the GPU consistently ran out of memory when testing the more resource-intensive models. Even after clearing storage to free up additional resources, these models failed to function properly, underscoring the significant storage and computational demands required for fine-tuning. These challenges highlight the limitations

Table 4: Performance Metrics for Various Models through fine-tuning. The best results are in **bold**. *Ran out of storage in GPU

Model	Accuracy	Recall	Precision	F1 Score	MCC
T5-base	0.9039	0.4286	0.6000	0.5000	0.4562
T5-small	0.9016	0.4490	0.5789	0.5057	0.4566
T5-large	**Resource limitations*				
Flan-T5-small	0.9359	0.4694	0.9200	0.6216	0.6307
Flan-T5-base	**0.9565**	0.6531	**0.9412**	0.7711	0.7632
Flan-T5-large	0.9268	0.3878	0.9048	0.5429	0.5644
Flan-T5-XL and XXL	**Resource limitations*				
GPT2	0.9428	0.7755	0.7308	0.7525	0.7206
GPT2-medium	**0.9565**	**0.7958**	0.8125	**0.8041**	**0.7797**
GPT2-large and XL	**Resource limitations*				

of using Google Colab's free-tier resources for such experiments.

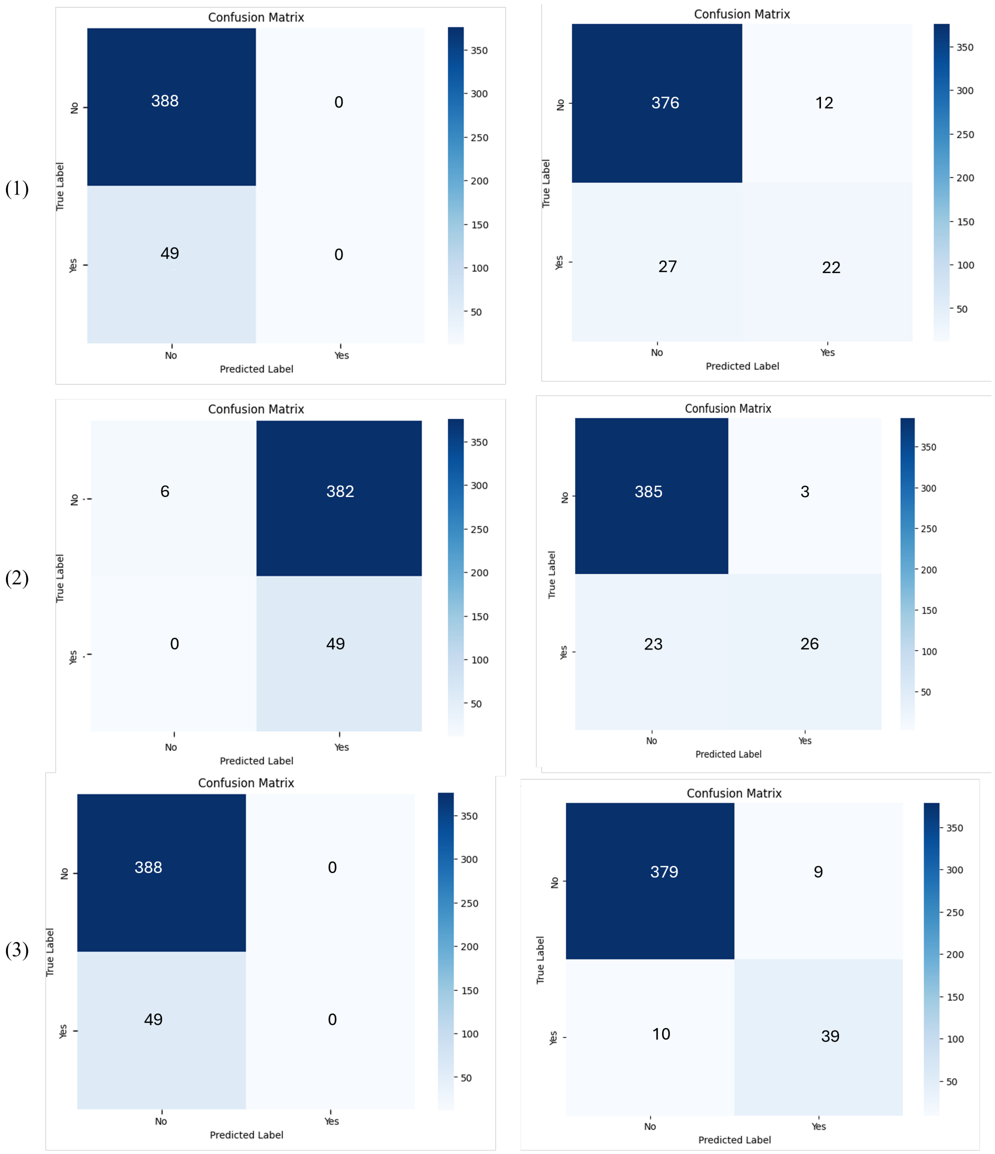
4.2 Experimental results

After the zero-shot experiments, some initial observations and conclusions could be made. The original hypothesis that larger LLMs with more complex architecture would have better performance metrics proved to be relatively inaccurate. Although the T5 models followed this trend with almost every performance metric, except for Precision and MCC, the GPT2 models (GPT2, GPT2-medium, and GPT2 XL) were similar in terms of their performance metrics. For these three models, every single performance metric was identical. Finally, the Flan-T5 models had shown better results with Flan-T5-XL with all the metrics except for Precision and MCC, which were the irregular performance metrics in the T5 models. It appeared that across all of the models, the Precision and MCC results seemed more surprising. Table 3 summarize the results obtained through the zero-shot learning experiments.

Overall, in looking at the results of all the models, the GPT2 models were the best and most consistent. Though T5-large seemed to have metrics similar, and sometimes even better, than the GPT2 models, the three T5 models were spread out and had huge disparities between them. These models improved rapidly as model architecture got more advanced, leading to excellent performance metrics in T5-large, but T5-small and T5-base were far behind other models in comparison. On the other hand, the GPT2 and Flan-T5 models were much closer in their performance metrics and were all consistently scoring high numbers. Between the two categories, GPT2 scored higher values in all of the metrics except for the MCC, as Flan-T5 had better correlations there. Across all three model categories, though, GPT2 came out of the zero-shot experiments with the best initial reaction.

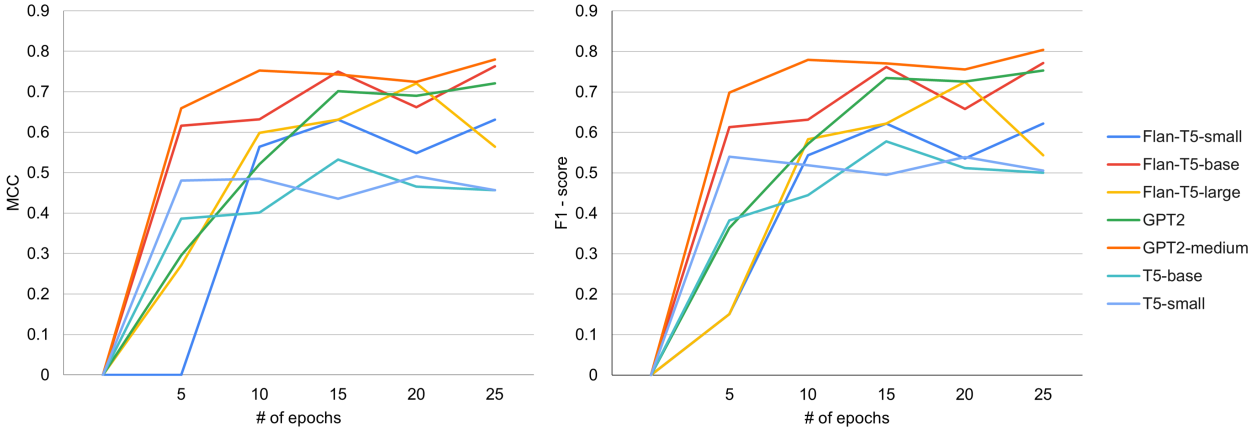
Fine-tuning further changed the results. Since the models needed much training to understand the task at hand and perform well, increases in the number of epochs were predicted to increase performance quality and help the models in training. For a majority of the models and performance metrics, this trend was accurate. There were multiple cases, in which the performance quality actually decreased slightly. For example, Flan-T5-small saw a decrease in its accuracy from 0.9359 to 0.9245 when the number of epochs increased from 15 to 20. However, most of the values appeared to increase with the epoch number.

Table 4 shows the values of each model's performance metrics at 25 epochs. Training each model for 25 epochs yielded the highest metric values, demonstrating their improved capability in identifying wildcard matches. Ultimately, the bolded results in Table 4 imply that GPT2-medium and Flan-T5-base were the best-performing models. LLMs that were more complex faced numerous problems during fine-tuning, so GPT2-medium and Flan-T5-base balance efficiency and performance the best among all of the models that were further tested.



(a) (b)

Figure 3: Confusion matrices illustrating the (1)T5-base, (2)Flan-T5-base, (3)GPT2-medium model's performance under (a) zero-shot learning and (b) fine-tuning.



(a) (b)

Figure 4: Fine-tuning performance comparison of the LLM in terms of (a) MCC and (b) F1 score.

5 Discussion

From the confusion matrices, as shown in Figure 3, it becomes clear that GPT2-medium, the best model, is accurate in identifying both true positives and true negatives, which are the cases that the model correctly labels. This improvement is made obvious by the fine-tuning confusion matrix. During the zero-shot GPT2-medium experiments, for instance, the model did not identify any true positives and labeled 49 cookies with a false label. However, after fine-tuning 25 epochs, the model could label 39 of the 49 cookies as true positives, showing massive leaps in reducing errors and learning the task at hand. Similar results were seen with Flan-T5-base, but the numbers of correctly labeled cookies were smaller than those of GPT2-medium. Only 26 wildcard matches were correctly labeled with a "Yes," and Flan-T5-base made 26 errors, which is more than GPT2-medium's 19 errors. Although T5-base was not among the top models after fine-tuning, it was the medium-sized model in the T5 section. Its confusion matrix results highlight the differences between reliable and unreliable models. It made 39 errors, far more than the other two models, and identified only 22 true positives. These disparities make GPT2-medium and Flan-T5-base the most improved and distinct models in the end.

Additionally, through the Figure 4 depicting the MCC and F1 trends, GPT2-medium remains the dominant model in both cases. Not only does GPT2-medium end with the highest scores in both cases, but it also consistently remains at the top of the graph for each epoch number, from 5 to 25. Because its statistics increased at almost every increment, excluding the 15 epochs case, GPT2-medium was the most reliable model. As expected, Flan-T5-base remained at the top of the graph, below GPT2-medium. It finished with the second-best statistics in both graphs, making it one of the best models after fine-tuning. In comparison, T5-base was consistently last or second-to-last in both graphs, further displaying huge differences between the accuracy and efficiency of the LLMs.

6 Conclusion

This work emphasized the importance of wildcard matches in overall browser and cookie security; it evaluated whether LLMs and AI could adapt to identify these matches well, benefiting the field of browser security.

By systematically evaluating GPT2, T5, and Flan-T5 models, we have shown that GPT2-medium and Flan-T5-base strike the best balance between efficiency and accuracy for wildcard match classification tasks, achieving up to **95%** accuracy and an MCC score of **0.77**.

Given the resource constraints encountered in fine-tuning larger models, it is clear that model selection must account not only for raw performance but also for computational feasibility. Future work may explore more efficient fine-tuning techniques (e.g., LoRA, parameter-efficient tuning) or distillation-based approaches to compress larger models. Additionally, integrating wildcard detection with other cookie security features—such as third-party blocking and advanced policy enforcement—could provide a more comprehensive cybersecurity strategy.

References

[1] K. Baker, “12 most common types of cyberattacks.,” 2024.

[2] N. C. S. Centre., “Mitigating malware and ransomware attacks.,” 2020.

[3] privacybee, “What is browser fingerprinting? here’s how to prevent it. what is browser fingerprinting? here’s how to prevent it.,” 2021.

[4] J. Estrada-Jiménez, J. Parra-Arnau, A. Rodríguez-Hoyos, and J. Forné, “Online advertising: Analysis of privacy threats and protection approaches,” *Computer Communications*, vol. 100, pp. 32–51, 2017.

[5] S. Englehardt and A. Narayanan, “Online tracking: A 1-million-site measurement and analysis,” in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 1388–1401, 2016.

[6] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[7] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.

[8] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz, “The web never forgets: Persistent tracking mechanisms in the wild,” in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pp. 674–689, 2014.

[9] T. B. Brown, “Language models are few-shot learners,” *arXiv preprint arXiv:2005.14165*, 2020.

[10] Y. Ye, T. Li, D. Adjeroh, and S. S. Iyengar, “A survey on malware detection using data mining techniques,” *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, pp. 1–40, 2017.

[11] A. C. Bahnsen, E. C. Bohorquez, S. Villegas, J. Vargas, and F. A. González, “Classifying phishing urls using recurrent neural networks,” in *2017 APWG symposium on electronic crime research (eCrime)*, pp. 1–8, IEEE, 2017.

[12] R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network intrusion detection,” in *2010 IEEE symposium on security and privacy*, pp. 305–316, IEEE, 2010.

[13] D. Chicco and G. Jurman, “The matthews correlation coefficient (mcc) should replace the roc auc as the standard metric for assessing binary classification,” *BioData Mining*, vol. 16, no. 1, p. 4, 2023.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998–6008, 2017.

[15] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, *et al.*, “Scaling instruction-finetuned language models,” *arXiv preprint arXiv:2210.11416*, 2022.

[16] B. Romera-Paredes and P. Torr, “An embarrassingly simple approach to zero-shot learning,” in *International conference on machine learning*, pp. 2152–2161, PMLR, 2015.

[17] Y. Xian, B. Schiele, and Z. Akata, “Zero-shot learning-the good, the bad and the ugly,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4582–4591, 2017.

[18] W. Wang, V. W. Zheng, H. Yu, and C. Miao, “A survey of zero-shot learning: Settings, methods, and applications,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–37, 2019.

[19] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving, “Fine-tuning language models from human preferences,” *arXiv preprint arXiv:1909.08593*, 2019.

[20] L. Xu, H. Xie, S.-Z. J. Qin, X. Tao, and F. L. Wang, “Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment,” *arXiv preprint arXiv:2312.12148*, 2023.