Automating the Cybersecurity Triage Process: A Comparative Study on the Performance of Large Language Models

Pascal Bakker
p.n.bakker@student.utwente.nl
University of Twente
The Netherlands

ABSTRACT

Security analysts have the task of inspecting cybersecurity alarms to filter false positives and identify their severity: triage. The problem with this process is that it is complicated and time-consuming, limiting the depth and speed of investigations. Whereas other proposed optimizations and automations appear to be very promising, rapid advancements in the development of Large Language Models (LLMs) opened up new possibilities to speed up parts of the triage process that previously required human judgment. This research aims to identify ways in which LLMs can optimize triage, evaluate the performance of these techniques and offer a comparison between different LLMs including GPT-4, Aya, Code Llama, Gemma, Llama 3, Mistral and Phi-3. The study shows that GPT-4 is the most capable model, while Llama 3 and Mistral achieve competitively similar results. The findings in this study are expected to help security teams make informed implementation decisions when optimizing the triage process. The data and scripts used are available in the GitHub repository (https://github.com/PascalNB/llm-triage-automation).

KEYWORDS

Cybersecurity, triage, analysis, large language model, natural language processing, automation

1 INTRODUCTION

The 2023 Cost of a Data Breach Report by IBM Security [15] concludes that the average cost of a security breach in 2023 was 4.45M USD, marking an increase of 2.3% since 2022 and a 15.3% increase compared to 2020. Only 1 in 3 breaches are identified by an organization's security team. However, organizations with high levels of incident response planning saved 1.49M USD on average, highlighting the pressing need for security investments in training, thread detection and response technologies.

One such investment is the use of Security Operation Centers (SOCs), which respond to security incidents in real-time. They consist of security analysts that investigate data from various sources such as Security Information and Event Management (SIEM) systems. The SIEM systems collect log data from a large number of sources such as network devices and applications within the organization's system. Based on rules, patterns and conditions, anomalies and suspicious activities are identified and alarms are created.

The number of alarms is immense, ranging from hundreds to thousands per day, of which a large portion are false positives or low priority. The volume and complexity of the alarms causes SOCs to miss serious attacks and inadvertently contributes to mistakes in the analysis. Besides that, it leads to security teams experiencing fatigue, and it contributes to internal friction and turnover [32].

Since the SOCs cannot respond to every single alarm, identifying the severity of alarms is an important step in the incident response workflow. This process, triage, involves understanding the impact of an alarm, correlating it with other alarms and identifying potential future goals of adversaries to conclude its severity. By prioritizing alarms, SOCs can focus their resources on high-severity alarms first, thus mitigating damages and reducing costs.

There are many proposed and implemented techniques to optimize triage and the SOC workflow. For example, Security Orchestration, Automation, and Response (SOAR) platforms have streamlined parts of the process by automating routine tasks, but many steps of triage still require human judgment to make adequate decisions [9]. Consequentially, the triage process is prone to human error. This, in combination with the volume and complexity of alarms, presses the need for the automation of triage.

The field of Artificial Intelligence (AI) has the potential to significantly impact the automation of triage. It involves the use of machines to perform tasks that mimic human actions such as reasoning, problem-solving and learning [33]. Machine Learning (ML) allows systems to solve problems by analyzing patterns in data and making predictions and decisions without explicit programming [34]. One subfield of ML is Natural Language Processing (NLP). NLP involves using computational approaches to process and transcribe natural-language texts with further goals such as translation, summarization, sentiment assessment or generation of texts. It plays a growing role in streamlining and automating business operations, and increasing productivity [14].

Large Language Models (LLMs) are a major advancement in the field of NLP. These models have been trained on immense amounts of natural-language data and are capable of understanding and generating texts to perform a wide range of tasks [13]. They are designed to be applied in any domain or industry, erasing the need to create or train a domain-specific ML model. Their ability to identify contextual relationships and recognize complex patterns [6] in a short amount of time makes LLMs the perfect entrypoint to automate triage and thus optimize the incident response workflow.

This research aims to explore the potential of LLMs in optimizing the triage process, establish ways to evaluate the performance of

TScIT 41, July 5, 2024, Enschede, The Netherlands

LLMs in cybersecurity, and present a comparison of different models when automating triage steps. To pursue this goal, the following research questions (RQ) provide the basis of this research:

RQ1: How can LLMs be integrated into the existing incident response workflow to streamline the triage process?
RQ2: What suitable evaluation metrics should be used to assess the performance of LLMs in cybersecurity triage?
RQ3: How do different LLMs compare in performance when optimizing the cybersecurity triage process?

This research is organized as follows:

Firstly, section 2 discusses literature about existing optimizations and identify tasks that can be optimized with LLMs. Secondly, section 3 determines which evaluation metrics are suitable to judge the performance of LLMs. Then, section 4 establishes the methodology set and evaluation framework. After that, section 5 discusses the results, implications and limitations of the study. Lastly, section 6 summarizes the key findings of this research.

2 OPTIMIZING TRIAGE USING LLMS

This section intends to answer RQ1: *How can LLMs be integrated into the existing incident response workflow to streamline the triage process?* Firstly, a rundown of the triage process is given by providing examples of the steps taken when identifying the priority of an alarm. Secondly, existing and proposed solutions of optimizing triage are summarized. Then, a general use of LLMs is given using a brief overview of the recent advances made in NLP, after which examples of LLMs performing relevant tasks are provided. Lastly, using these findings, possible automations in the triage process are identified.

2.1 Steps of Triage

The goal of triage is to follow a structured process to quickly assess and prioritize security alarms. Although the steps of triage are not set in stone and depend on the type of alarm, there are a number of basic tasks that can be followed. Based on consultations with security analysts, the following tasks can be identified:

(1) **Understanding the alarm:** The security analyst intends to understand the nature of the alarm. This includes the origin of the alarm (e.g., in the cloud or on-premise) and what time it was created. Besides this, the analyst reviews any prior communications that suggest the possibility of alarm creation, and assesses its relevance to the alarm in question.
(2) **Analyzing the context:** The analyst looks through the given alarm data and identifies the affected entities and to what extent they are affected. This includes users, data, systems and operations.
(3) **Correlating the alarm:** Based on the alarm context, the analyst searches whether the alarm has been previously encountered or if related alarms have occurred, either within the affected network or a different one.
(4) **Identifying the position in the kill chain:** To assess potential consequences, the position of the alarm within the kill chain is determined. A kill chain describes the stages of a cyberattack, from initial reconnaissance to the final goal. This is done by referencing the MITRE ATT&CK [42] or Cyber Kill Chain [27] frameworks. This also depends on other potentially correlated alarms.
(5) **Prioritizing the alarm:** The analyst concludes how severe the alarm is and assigns a priority of high, medium, low or no threat. The priority classification determines how quickly an alarm requires further analysis and response actions.

The overall process should ideally not take longer than 30 minutes, because high-priority alarms need to be handled as urgently as possible. To meet this time constraint, it is essential to integrate tools and processes to allow analysts to perform triage efficiently.

2.2 Existing Triage Automations/Optimizations

One goal of triage is to correlate alarms, which entails identifying if the alarms are related to determine their placement within the kill chain. Ficke [12] optimizes this step by using alert trees. These trees are data structures used to organize and visualize generated alerts. The alerts are structured hierarchically, showing the relationships between alerts and the sequence of events. The proposed solution also eliminates redundancies in the graphs, thereby preventing the graphs from reaching sizes consisting of thousands of nodes. Based on academic datasets, the result is a system that quickly reconstructs paths that give insight into multistep threats in a network, which is an otherwise time-consuming task for human analysts.

Additionally, Serketzis et al. [41] propose a model that integrates Cyber Threat Intelligence (CTI) into the process. CTI involves collecting and analyzing information about potential and existing threats. The model aims to enhance Digital Forensic Readiness (DFR), which is the preparation of digital forensics through collection and storage of data, to create relevant readily available information. Three independent but interrelated modules form the basis of the model:

- **IoC Collection Module:** Indicators of Compromise (IoC) consist of indicators of malicious activity observed in networks or systems. The module aggregates IoC from many internal and external sources of CTI to increase the collective knowledge. Data is evaluated and correlated to further increase its value, after which it is kept in a database.
- **Audit Log Processing Module:** This module aims to gather, validate and process audit log data generated by different parts within the organization. The data is stored in a dedicated database, which handles retrieval requests from the third module.
- **Threat Identification Module:** This module cross-matches the contents produced by the other modules and identifies threats. The evidence is stored in Intelligence Evidence Storage Systems (IESSs), which act as an entry point for analysts looking to preview potentially adverse incidents. Besides identifying suspicious activity, the module provides potential instigating factors.

The resulting model is shown to have a high accuracy of 90.73% when testing network data for malicious activity.

Besides that, Zhong et al. [48] approached the automation of triage by tracing the operations of professional security analysts. As junior analysts are typically responsible for conducting triage, this approach effectively speeds up the process. Finite state machines were constructed based on the senior analysts' patterns and were used to achieve high-speed triage with a low number of false positives. However, limitations include a high number of false negatives as well as a dependency on high-performing security analysts to maximize the performance of the automated system. Extending this approach, Lin [25] feeds contexts into a recurrent neural network which detects matching traces and presents these to novice analysts which in turn trains them in effective triage.

Finally, it is worth noting that a high level of automation can have adverse effects on the overall performance of security analysts when performing triage because of the following reasons:

- Understanding and conducting full-time monitoring of the automation can increase workload [19].
- The level of trust in automation can lead to over-reliance or negligence [22].

Hence, it is crucial to maintain a level of human interaction when automating the steps of triage, and to prioritize user-friendliness of integrated tools.

The limitation of all the previously proposed automations is that they do not involve NLP. Triage requires understanding and interpreting content surrounding the alarm, such as logs, announcements and other forms of natural or unstructured language, which are challenging to automate through conventional methods. However, LLMs can be the entry point to find solutions to automate such tasks.

2.3 General Usage of LLMs

LLMs make use of Natural Language Understanding (NLU) and Natural Language Generation (NLG). NLU aims to comprehend meaning and intent in natural language, while NLG focuses on generating original human-like texts. To achieve NLU and NLG, language models make use of so-called encoders and decoders. The purpose of encoders is to turn input texts into fixed-size vectors that act as abstract representations. Language models then utilize decoders to transform such representations into a generated target output. This approach works well on tasks that map input sequences to output sequences (sec2sec), such as language translation [8, 43].

In 2014, Bahdanau et al. [4] introduced the concept of attention which rids the encoders of creating fixed-length vectors. It allows language models to focus on the most relevant parts of texts and enabling operations on much longer input sequences. Based on this, Vaswani et al. [46] developed the transformer architecture in 2017, setting the precedent for modern LLMs. Transformers are superior in quality, more parallelizable, and take less time to train.

Early well-known examples of such transformer-based models are BERT (Bidirectional Encoder Representations from Transformers) [11], and GPT (Generative Pre-trained Transformer) [36]:

- BERT was specifically designed as a pre-trained model to be easily fine-tuned for a wide range of tasks such as answering questions and natural language inference, without the need of task-specific architecture. In the cybersecurity domain, BERT models have been fine-tuned to detect malicious software [38] and phishing emails [23], in addition to performing general cybersecurity tasks [5].
- GPT is pre-trained on a large amount of unlabeled data and designed to generate coherent context-specific text. It works by generating continuations based on the input text through probabilistic guesses. Like BERT, it requires fine-tuning to adapt the model to specific tasks.

The disadvantage of these models is that they are relatively small and require fine-tuning to be applied to domain-specific tasks. Fine-tuning using domain-specific data is not only resource and time-intensive, but the resulting models have limited applicability and can potentially include bias. Another disadvantage is the phenomenon of catastrophic forgetting where existing models forget their original knowledge after being trained on new data.

The introduction of much larger general models such as GPT-4 [2] and Llama 3 [28] intends to eliminate these problems. The broad availability and applicability of these large general models allows organizations to easily integrate them to optimize general and specific tasks. These models are relatively new and currently have limited research available on their use, but there are some notable examples.

Nori et al. [29] have used GPT-4 to assess its performance on medical tests without fine-tuning. The result shows that the model passed the tests without specialized instructions, even surpassing other models that are fine-tuned on medical knowledge. This shows that the significance of fine-tuning decreases when general models increase in size and capabilities.

In the cybersecurity domain, research on general models is sparse, but applications of fine-tuned LLMs still provide valuable insights into their potential for optimizing triage. For example, Karlsen et al. [20] propose a system that uses LLMs to perform log analysis. Where previous methods of analysis relied on rule-based or statistical approaches, LLMs are able to learn complex patterns and relationships within log data without requiring manual feature engineering. The study focuses on fine-tuning models such as BERT and GPT-2 [37] through self-supervised learning where the LLMs automatically label data by identifying relations. Larger models like GPT-4 and Llama 2 [45] were not used due to their high parameter counts, leading to increased computational requirements during the fine-tuning process.

To instruct LLMs to perform tasks, prompts are used. They consist of textual inputs that the model interprets and uses to generate contextually relevant responses. Prompts are split into system prompts and user prompts. System prompts act as predefined inputs that guide the model's behavior and control the context of a task. User prompts are usually provided by end-users and describe the specific task. They are diverse in form and content. Creating good prompts is essential to achieve high-quality LLM outputs. Short and simple prompts might lack sufficient context, leading to incomplete and inaccurate results. On the other hand, excessively long and complex prompts might introduce redundant information that confuses the model. Therefore, balancing the sizes and complexities of prompts is crucial to obtain relevant, accurate and coherent outputs.

2.4 Using LLMs to Optimize Triage

When applying LLMs to specific tasks, it is important to keep the tasks short and simple to ensure the results are consistent and easily testable. Referring to the triage steps in 2.1, all tasks that involve natural language or unstructured textual data have the potential to be automated using LLMs. Since the optimizations are ideally applicable across all kinds of alarms, the following possible automations are identified:

- **Detecting if an email is an announcement:** This means letting the LLM process the entire email and concluding if it contains any information regarding actions that could trigger alarm generation.
- **Detecting if an announcement is related to an alarm:** This involves feeding both an announcement and alarm data into the LLM and determining whether a correlation exists, meaning the alarm was triggered by an action that was announced beforehand.
- **Correlating an alarm with other alarms and identifying if there are relationships:** Based on the alarm data, potentially related alarms are collected, after which the LLM concludes if they are in fact correlated.
- **Determining the position of an alarm in the kill chain:** Based on the MITRE ATT&CK framework, alarms have a position in the kill chain. An LLM can use alarm data and correlated alarms to identify this position.
- **Determining the priority of an alarm as high, medium, low or no threat:** This last step is to use the answers of the previous tasks to determine if the alarm should be treated as high, medium, low or no priority.

In summary, incorporating LLMs into the incident response workflow by automating these steps will allow for a more efficient triage process. By automatically detecting if an alarm was generated due to a previously announced action, it can be concluded whether the alarm is benign and if it poses no actual threat. This will reduce the number of false positives that an analyst has to investigate.

Besides that, automatically correlating alarms and concluding the kill chain position and priority will reduce the workload of an analyst. This will streamline the triage process and thus prevent overloading alarm queues.

3 EVALUATION OF LLMS IN CYBERSECURITY TRIAGE

This section intends to answer RQ2: *What suitable evaluation metrics should be used to assess the performance of LLMs in cybersecurity triage?* Firstly, existing evaluation metrics for LLMs are identified. Finally, the most suitable metrics are determined to establish a testing framework for LLMs in the context of cybersecurity triage-related tasks.

3.1 Existing LLM Evaluation Metrics

Due to the inherent ambiguity of human language, it is challenging to evaluate the output of an LLM. Outputs of LLMs are not numerical in nature, but evaluation algorithms should produce a numerical score. This necessitates the use of sophisticated evaluation metrics.

3.1.1 Statistical-based evaluation. Besides simple human evaluation techniques like expert reviews and crowdsourcing, there are some notable automated metrics to measure LLM performance:

- The BLEU [35] score is specifically designed to test machine translation by matching output texts with reference texts.
- The ROUGE [24] score is used to evaluate text summaries by comparing model outputs with expected outputs.

These evaluation scores are purely statistical and thus reliable, but do not consider the nuances of semantics. They demonstrate a low correlation with human judgments, particularly in tasks related to creativity and diversity [26].

3.1.2 NLP-based evaluation. NLP-based evaluation techniques are more accurate but less reliable due to factors such as randomness, bias, and its inherent dependency on training data. Metrics such as BERTScore [47] and BLEURT [40] use descriptive LLMs such as BERT to provide a score by comparing generated and reference texts while taking semantics into account.

Besides that, Liu et al. [26] propose G-EVAL, a framework that uses generative LLMs such as GPT-4 or GPT-3 [7] to evaluate LLM outputs. First, evaluation steps are generated based on a given task and evaluation criteria. Then, the steps are used to assess an LLMs output and a score ranging from 1 to 5 is given. The resulting score takes semantics into account, and the resulting evaluation is more correlated with human judgment. However, it is unreliable due to the arbitrary nature of LLM output, and it is biased towards LLM-generated texts compared to human-written texts.

Similarly, Kim et al. [21] introduce Prometheus, an LLM-based evaluator. Proprietary LLMs, such as used in G-EVAL, do not fully disclose internal operations, limiting fair evaluations. Furthermore, they might force version updates, impacting the consistency and replicability of evaluations. Lastly, financial constraints can make their use challenging. Because of this, Prometheus uses an open-source LMM that is fine-grained for evaluation and on par with the evaluation performance of GPT-4.

3.1.3 Score-based evaluation. The assigned automations in the triage process are task-specific and only require tests on the correctness of the LLM's answer. Answers are classified as true positive, false positive, true negative and false negative, depending on the data's actual classification and the model's prediction. Using these four classifications, different suitable task-specific evaluation metrics can be constructed:

- **Accuracy:** The ratio of correct predictions to the total number of answers. It gives an overall indication of the model's ability to make correct predictions, but can be misleading if the testing or real-world data is imbalanced.
- **Precision:** The ratio of correct answers compared to all answers that were flagged positive by the LLM. A high precision indicates a low false positive rate.
- **Recall:** The ratio of correct predictions to the total number of actual positives in the test data. A high recall indicates a low false negative rate.
- **F1-score:** A harmonic mean of precision and recall. As a metric, it represents a balance between precision and recall, capturing the performance using both metrics.

3.1.4 Efficiency-based evaluation. Besides answer-based evaluation techniques, the following additional metrics can be identified to assess the efficiency of an LLM when performing a task:

- **Median time:** The median amount of time it takes for the LLM to provide a response. While not as important as the

previous metrics, it provides an insight into the responsiveness of the model. This is applicable in cases where quick responses are critical, such as in the incident response workflow. The median time is preferred over the mean time to mitigate the effect of outliers.
- **Output Token Count:** The number of tokens in the output produced by the LLM. It provides an insight into both the computational and financial costs of processing a prompt. It is particularly useful when comparing different models, because some models may generate unnecessarily verbose responses. However, it has no significant use when all models are prompted to adhere to a specific output format such as a single value or in JSON.

From this, a **cost-efficiency** metric can be introduced that evaluates the model's performance relative to the cost incurred. For example, a model with a high performance score should not be preferred if the token costs make integrating the model financially infeasible.

3.1.5 Reliability-based evaluation. In addition to performance metrics, it is important to verify the reliability of models by ensuring that they consistently produce outputs in the expected formats. The following performance metrics are suitable to measure these characteristics:

- **Consistency:** Evaluates whether the model provides consistent output for the same inputs. LLMs generally incorporate a degree of randomness to prevent training overfitting and ensure the diversity of results. However, this might not be beneficial when consistent outputs are crucial. Moreover, a low consistency could decrease the reliability and replicability of the evaluation.
- **Error rate:** The rate of incorrect responses. For example, the rate of outputs that do not follow an explicit format.

3.2 Applying Evaluation Metrics to Triage

In the triage process, a large number of false positives would cause the alarm queue to be filled up, resulting in limited time for analysts to conduct thorough investigations. On the other hand, a large number of false negatives would result in critical alarms being missed. Therefore, it is important to balance these metrics when evaluating LLMs. In the context of tasks that require consistent and strictly formatted outputs, the F1-score will provide a suitable performance score that balances these concerns.

When dealing with tasks that have definitive right or wrong answers, accuracy is the most appropriate measure to assess performance. This is also the case for multiclass classifications that are relatively balanced. For instance, the task of identifying the MITRE ATT&CK tactic or technique associated with an alarm or action can be evaluated through accuracy.

For all tasks, the error rate is an important metric. It assesses the general usability of a model, because models are challenging to evaluate and impossible to implement within an organization when they frequently produce unusable responses. Additionally, the consistency of a model is important because it ensures that an evaluation reflects the real-world implementation.

Incorporating these metrics into an evaluation framework provides a suitable procedure to assess the performance of LLMs in cybersecurity triage.

4 COMPARING LLMS IN CYBERSECURITY TRIAGE

This section intends to answer RQ3: *How do different LLMs compare in performance when optimizing the cybersecurity triage process?* The automation steps in section 2.4 are combined with the evaluation metrics as described in section 3.2 to perform a comparison between various LLMs acros different tasks. Firstly, an overview of the experiment setup is given. After that, the comparison results are provided.

4.1 Comparison Framework and Setup

The first LLM included in the comparison is GPT-4. The other models selected are openly available and accessible through the Ollama [30] library. The Ollama language model platform is chosen for its ease of use and its simplicity to deploy the most recent state-of-the-art models. It is capable of running a local server that receives API calls, enabling the execution of operations on models from external applications like a Jupyter Notebook.

Most models are released as a collection of multiple variants, each with a different parameter count. A model with a large number of parameters is expected to perform better, but requires more computational resources in contrast to a small model. Due to computational limits, only models with parameter counts of up to 14 billion have been selected. The selected models are Aya 23 [3], Code Llama [39], Gemma [44], Llama 3, Mistral [18] and Phi-3 [1]. The different models and their sizes and characteristics are given in Table 1. The Ollama models use a 4-bit quantization as is provided by default.

The evaluation framework is written in Python and contained within a Jupyter Notebook. This supports transparency and replicability of the experiment because the code can be executed on other future LLMs to assess and compare their performances.

For each task, prompts of different lengths are constructed to assess the impact of prompt size on the model's performance. Complex tasks are not possible to execute with short prompts due to the lack of sufficient information.

The data used in this study consists of 20 cybersecurity announcement emails provided by Northwave Cyber Security. The emails are manually analyzed to determine what possible alarms could be generated as a consequence of the announced actions, after which the MITRE ATT&CK tactics are identified. For example, an email announcing that a certain person will add accounts as admins to local and production servers is assigned the following tactics: 1) privilege escalation: accounts are given higher permissions on a system; 2) persistence: maintaining high-level access by modifying permission groups ensures persistence; 3) initial access: logins through valid accounts could indicate initial access.

Additionally, 20 non-announcement emails are randomly taken from the Enron email dataset [10] to construct a joined dataset of 40 labeled emails. The code of the framework and the used data and prompts are available on GitHub (https://github.com/PascalNB/llm-triage-automation).

Table 1: Selected models and their characteristics. The parameter count of GPT-4 is made *italic* because it is an estimation.

Model	Parameters	Description
GPT-4	*1760B*	Developed by OpenAI as their fourth generation in their GPT series. It is estimated to have 1.76 trillion parameters.
Phi-3	14B	A lightweight open model developed by Microsoft tailored to logic reasoning.
Code Llama	13B	A fine-tuned Llama model by Meta and designed for generating and discussing code.
Aya 23	8B	A multilingual model developed by Cohere that supports 23 languages.
Llama 3	8B	A capable pre-trained model developed by Meta and the third in their Llama series.
Gemma	7B	Part of a family of lightweight models developed by Google DeepMind.
Mistral	7B	A versatile model developed by Mistral AI.
Phi-3	3.8B	The smallest model variant in the Phi-3 series.
Gemma	2B	The smallest model variant of the Gemma model family.

Using this data, the following tasks are evaluated:

(1) **Announcement Detection:** The LLM has to determine whether a given email is a cybersecurity announcement. The task is executed using three separate system prompts: (a) A long prompt that explains the task, gives examples of security actions and requests a JSON output. (b) A medium-sized prompt that explains the task and briefly gives some examples of security actions. (c) A short prompt that briefly explains the task.
All three prompts request the model to output its answer in JSON with a single key `is_announcement`. Finally, the F1-score, median time and error rate are recorded as evaluation metrics.

(2) **Tactic Detection:** The LLM has to determine what the MITRE ATT&CK tactic is of the alarms that could be generated as a result of the activity that is expressed in an announcement. These tactics are: Reconnaissance, Resource Development, Initial Access, Execution, Persistence, Privilege Escalation, Defense Evasion, Credential Access, Discovery, Lateral Movement, Collection, Command and Control, Exfiltration, and Impact. The task is executed using two different system prompts: (a) A long prompt that explains the context, gives a list of the existing tactics and requests the model to predict the tactic of potentially generated alarms. (b) A medium-sized prompt that explains the context, gives a list of the existing tactics and requests the model to give the tactic of the action expressed in the announcement.
Both prompts request the model to output its answer in JSON with a single key `tactic`. The resulting evaluation metrics are accuracy, median time and error rate.

Besides requesting a JSON response, both OpenAI and Ollama models can be formally instructed to exclusively output JSON, ensuring that their outputs adhere to the correct format. However, this does not cover which keys should be included in the final response, as those should be included in the prompt itself. A response that lacks the requested keys will be considered an error and affects the model's error rate.

Finally, all evaluations are conducted within the Jupyter Notebook running with an AMD Ryzen 7 4800H CPU and an NVIDIA GeForce RTX 2060 GPU.

5 RESULTS AND DISCUSSION

This section will give an overview of the results and highlight the key findings on the performance of LLMs in cybersecurity tasks, as well as identify the limitations of this study. The results show that models like GPT-4, Llama 3, Mistral and Phi-3 3.8B excelled in detecting cybersecurity announcements, particularly with long prompts. GPT-4 also demonstrated the highest accuracy in identifying MITRE ATT&CK tactics.

The task of detecting cybersecurity announcements was performed using the three prompt sizes on all nine models. An overview of the results is presented in Table 2.

For the long prompt, GPT-4, Code Llama, Llama 3, Mistral and Phi-3 3.8B scored equally high with an F1-score of 1. GPT-4 also scored high with the medium prompt with an F1-score of 0.95, which Phi-3 3.8B and Llama 3 closely followed with scores of 0.927 and 0.9 respectively. Llama 3 scored the highest on the task with the short prompt with a score of 0.837.

The task of detecting the MITRE ATT&CK tactic of potential alarms following an email announcement was performed using the two prompt sizes on all nine models. An overview of the results is presented in Table 3.

GPT-4 showed the highest accuracy for both the long and medium-sized prompts with scores of 0.85 and 0.8 respectively. Mistral was generally the second-best model with accuracies of 0.7 and 0.6. Llama 3 followed this with scores of 0.6 and 0.65. All models performed error-free results, which is why error rates are excluded from the table.

Firstly, it is important to note that GPT-4 is the only model not operating within the same system as the other models. Changes in median times could be attributed to increased network traffic, server requests or timeouts, but the exact causes are unknown. Hence, time-based metrics should generally only be compared amongst models within the same system. However, the median times still give an indication of expected evaluation durations when incorporating GPT-4 to perform similar tasks.

For all tasks of the conducted experiment, the median times of most models remained consistent regardless of prompt size. This suggests that prompt size does not have a significant impact on the evaluation time. Therefore, the choice of best prompt is dependent only on the resulting performance score.

Considering the simple task of detecting announcements, only Llama 3, Mistral and Phi-3 3.8B showed relatively consistent high

Table 2: F1-scores, median evaluation times in seconds and error rates of each model for different prompt sizes when performing announcement detection. The best statistics are highlighted in bold and the next-best statistics are underlined. The parameter count of GPT-4 is made *italic* because it is an estimation. The median times of GPT-4 are made *italic* because it is the only model operating within a different system. Error rates of 0 are excluded.

Model	Parameters	Long prompt			Medium prompt			Short prompt		
		F1-score	Median time	Error rate	F1-score	Median time	Error rate	F1-score	Median time	Error rate
GPT-4	*1760B*	**1.000**	*8.456*	-	**0.950**	*7.541*	-	0.621	*10.940*	-
Phi-3	14B	0.842	3.946	0.025	0.452	2.952	0.150	0.600	2.974	0.750
Code Llama	13B	**1.000**	3.426	-	0.706	3.389	-	0.529	3.315	-
Aya 23	8B	0.844	1.513	-	0.809	1.421	-	0.516	1.394	-
Llama 3	8B	**1.000**	0.656	-	0.900	0.619	-	**0.837**	0.623	-
Gemma	7B	0.710	1.787	-	0.710	1.687	-	0.684	1.680	-
Mistral	7B	**1.000**	0.487	-	0.895	0.411	-	0.743	0.405	-
Phi-3	3.8B	**1.000**	0.355	-	0.927	0.375	-	0.706	0.875	-
Gemma	2B	0.788	0.194	-	0.333	0.192	-	0.667	0.192	-

Table 3: Accuracies, median evaluation times in seconds and error rates of each model for different prompt sizes when identifying MITRE ATT&CK tactics of potential alarms following an announcement. The best statistics are highlighted in bold and the next-best statistics are underlined. The parameter count of GPT-4 is made *italic* because it is an estimation. The median times of GPT-4 are made *italic* because it is the only model operating within a different system.

Model	Paremeters	Long prompt		Medium prompt	
		Accuracy	Median time	Accuracy	Median time
GPT-4	*1760B*	**0.850**	*3.543*	**0.800**	*1.765*
Phi-3	14B	0.400	4.433	0.400	3.957
Code Llama	13B	0.450	3.793	0.550	3.823
Aya 23	8B	0.350	2.024	0.350	1.956
Llama 3	8B	0.600	0.789	0.650	0.775
Gemma	7B	0.500	1.990	0.550	1.985
Mistral	7B	0.700	0.464	0.600	0.504
Phi-3	3.8B	0.400	0.900	0.300	0.890
Gemma	2B	0.250	0.212	0.300	0.217

scores across all three prompt sizes. Although GPT-4 had the best scores for long and medium-sized prompts, it fell behind on the short prompt. In general, not only did smaller models operate faster than larger models, but Llama 3 and Mistral produced overall better results across all tasks and prompts than bigger models like Code Llama and Phi-3 14B. Consequently, Llama 3, Mistral and Phi-3 3.8B are good choices when automating this simple task, as well as GPT-4 if processing times and its proprietary nature are of no concern when integrating a model into an organization.

Notably, Phi-3 14B is the only model that struggled to adhere to the requested output format with error rates of 0.03, 0.15 and 0.75 for long, medium and short prompts respectively. All errors can be attributed to the model's frequent inability to spell `is_announcement` correctly. This makes the model unreliable for simple tasks such as announcement detection.

On the task of detecting the MITRE ATT&CK tactic of potentially generated alarms following a cybersecurity announcement, only GPT-4 displayed a level of competence. Although Llama 3 and Mistral had the second-best statistics, their accuracies can still be considered insufficient, because an incorrect categorization for at least 30% of the time can have drastic effects within the incident response workflow. All other models demonstrated a low accuracy, which highlights the difficulty in providing a comprehensive description of the task within a single prompt. Models like Llama 3 and Mistral should thus only be considered in cases where a short evaluation time is critical or when an organization strives only to use openly available models.

Another noteworthy finding is that the accuracy does not significantly differ between prompt sizes. One reason for this could be that the medium prompt is not of sufficient quality and the long prompt is of equal quality but contains additional redundant information, resulting in equally low accuracies. However, GPT-4's ability to accurately perform the task is a counterexample to this, which means that the other models are simply too small or require more training or fine-tuning. For example, smaller models might not have been trained on the required MITRE ATT&CK data and thus lack the knowledge to detect tactics. Although many papers conclude that fine-tuning yields higher performances, they do not include recent LLMs such as GPT-4 as a comparison. Besides that, fine-tuning is out of scope for this study, which focuses solely on the performance of general LLMs.

5.1 Limitations

This study faces several limitations that impact the generalizability of the findings. Firstly, the datasets of 40 emails for announcement detection and 20 emails for tactic detection are small. These sizes limit the robustness of the results, which might not accurately reflect real-world scenarios. To draw more definitive conclusions about the models' performances, larger and more diverse datasets are needed.

Secondly, the absence of actual alarm data hindered the ability to conduct an evaluation for other tasks that are critical in cybersecurity triage. This gap limits the scope of the assessment, because organizations might still desire a comparison of LLMs when automating these tasks.

Besides that, hardware limitations restricted the study to relatively small models, because larger and more powerful models had substantial computational requirements. This means the study might not reflect the full potential of general LLMs. However, the assessment of the included models is still valuable for research into smaller LLMs and environments with limited resources.

Additionally, this research is only focused on general LLMs and excludes fine-tuned models which could yield higher performances on specific tasks. Despite this, the results are still valuable to organizations which do not have the resources to create custom fine-tuned models.

Lastly, there is no formal separation between triage and further alarm analysis, because the only requirement in the process is assigning a priority. The identified steps of triage are based on examples and consultations with security analysts, but might not be applicable in all alarm contexts or SOC environments.

Despite limitations, this study still provides a baseline for understanding how LLMs can be used to automate tasks in cybersecurity triage, an evaluation framework to assess and compare LLMs performances, and an overall comparison of available general LLMs. The insights in this study pave the way to future research with other tasks, models and datasets, as well as guiding organizations with practical implementations when optimizing triage.

5.2 Future Works

Avenues for further studies include using the given evaluation and comparison framework to offer more extensive comparisons between LLMs. Studies could include LLMs that are specifically fine-tuned for cybersecurity or other large proprietary models, as well as the larger counterparts of the already-included models like Llama 3 and Mistral.

Additionally, future works can include alarm data to automate and evaluate tasks such as alarm correlation, kill chain position identification and priority categorization. This results in a more comprehensive assessment of LLMs' capabilities in handling real-world incidents.

Other research could also explore how triage automation affects user interaction with incident response systems, if optimizations do decrease the strain on security analysts, and ultimately if it results in fewer security incidents and reduced costs and damages.

6 CONCLUSION

This study aimed to investigate the integration of LLMs to optimize the cybersecurity triage process and offer a comparison of various LLMs through a systematic evaluation. Where proposed and existing optimizations show promising results, many steps in triage still require analyzing and understanding natural language. A literature review has shown that general LLMs are capable of proficiently executing these steps. Although fine-tuned models have been the stat-of-the-art solution for specific tasks, recent general LLMs have shown a comparable competence.

After identifying suitable performance metrics, comparative analysis of different LLMs revealed that GPT-4 consistently performed at a high level when completing cybersecurity announcement detection and MITRE ATT&CK tactic classification. Llama 3 and Mistral showed a competitive performance on all tasks, while the 3.8B parameter version of Phi-3 achieves good results on simple tasks.

Despite limitations on the amount and type of data used, this research provides a comprehensive framework for evaluating LLMs in cybersecurity triage. Further research is needed to analyze the performance of LLMs in other parts of the triage process, and establish a comparison including fine-tuned or larger models.

DISCLAIMER ON THE USE OF AI

During the preparation of this work, the author used generative AI tools such as ChatGPT [31], Llama 3, JetBrains Grazie [16] and code completion in JetBrains PyCharm [17] for the following purposes:

- Find definitions of terms and concepts when conventional tools and search engines are unsatisfactory.
- Check and correct the spelling of words and grammar of sentences.
- Improve the readability of sentences and paragraphs through rewording and restructuring.
- Use code completion functionality to speed up programming tasks.

After using these tools/services, the authors reviewed and edited the content as needed and take full responsibility for the content of the work. The services were not used to produce scientific insights, create figures, draw conclusions or provide recommendations.

REFERENCES

[1] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219* (2024).

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[3] Viraat Aryabumi, John Dang, Dwarak Talupuru, Saurabh Dash, David Cairuz, Hangyu Lin, Bharat Venkitesh, Madeline Smith, Kelly Marchisio, Sebastian Ruder, et al. 2024. Aya 23: Open Weight Releases to Further Multilingual Progress. *arXiv preprint arXiv:2405.15032* (2024).

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[5] Markus Bayer, Philipp Kuehn, Ramin Shanehsaz, and Christian Reuter. 2024. Cysecbert: A domain-adapted language model for the cybersecurity domain. *ACM Transactions on Privacy and Security* 27, 2 (2024), 1–20.

[6] Sebastian Bordt, Ben Lengerich, Harsha Nori, and Rich Caruana. 2024. Data Science with LLMs and Interpretable Models. *arXiv preprint arXiv:2402.14474* (2024).

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[8] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[9] Anton Chuvakin. 2019. *The Uphill Battle of Triaging Alerts*. Dark Reading. https://www.darkreading.com/threat-intelligence/the-uphill-battle-of-triaging-alerts

[10] Enron Corp and William Cohen. 2015. *Enron Email Dataset*. https://www.loc.gov/item/2018487913/

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[12] Eric Ficke. 2022. *Reconstructing Alert Trees for Cyber Triage*. Ph. D. Dissertation. The University of Texas at San Antonio.

[13] IBM Corporation. n.d.. *What are large language models (LLMs)?* IBM Corporation. https://www.ibm.com/topics/large-language-models

[14] IBM Corporation. n.d.. *What is natural language processing (NLP)?* IBM Corporation. https://www.ibm.com/topics/natural-language-processing

[15] IBM Security. 2023. Cost of a Data Breach Report 2023. https://www.ibm.com/reports/data-breach

[16] JetBrains. 2024. JetBrains Grazie: AI-based writing and communication companion for people in tech. https://lp.jetbrains.com/grazie-for-software-teams/

[17] JetBrains. 2024. *PyCharm*. https://www.jetbrains.com/pycharm/

[18] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023).

[19] David B Kaber and Mica R Endsley. 2004. The effects of level of automation and adaptive automation on human performance, situation awareness and workload in a dynamic control task. *Theoretical issues in ergonomics science* 5, 2 (2004), 113–153.

[20] Egil Karlsen, Xiao Luo, Nur Zincir-Heywood, and Malcolm Heywood. 2024. Large language models and unsupervised feature learning: implications for log analysis. *Annals of Telecommunications* (2024), 1–19.

[21] Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, et al. 2023. Prometheus: Inducing fine-grained evaluation capability in language models. *arXiv preprint arXiv:2310.08491* (2023).

[22] John D Lee and Katrina A See. 2004. Trust in automation: Designing for appropriate reliance. *Human factors* 46, 1 (2004), 50–80.

[23] Younghoo Lee, Joshua Saxe, and Richard Harang. 2020. CATBERT: Context-aware tiny BERT for detecting social engineering emails. *arXiv preprint arXiv:2010.03484* (2020).

[24] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.

[25] Tao Lin. 2018. A Data Triage Retrieval System for Cyber Security Operations Center. (2018).

[26] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. Gpteval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634* (2023).

[27] Lockheed Martin. 2011. Cyber Kill Chain. https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

[28] Meta. 2024. Introducing Meta Llama 3: The most capable openly available LLM to date. https://ai.meta.com/blog/meta-llama-3/

[29] Harsha Nori, Nicholas King, Scott Mayer McKinney, Dean Carignan, and Eric Horvitz. 2023. Capabilities of gpt-4 on medical challenge problems. *arXiv preprint arXiv:2303.13375* (2023).

[30] Ollama. 2023. *Ollama*. https://github.com/ollama/ollama

[31] OpenAI. 2023. ChatGPT: A Large Language Model. https://www.openai.com/chatgpt

[32] Orca Security. 2022. 2022 Cloud Security Alert Fatigue Report. https://orca.security/wp-content/uploads/2022/03/Orca-2022-Cloud-Security-Alert-Fatigue-Report.pdf

[33] Oxford English Dictionary 2024. artificial intelligence, n. In *Oxford English Dictionary*. Oxford University Press. https://doi.org/10.1093/OED/3194963277

[34] Oxford English Dictionary 2024. machine learning, n. In *Oxford English Dictionary*. Oxford University Press. https://doi.org/10.1093/OED/2166790335

[35] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

[36] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. (2018).

[37] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[38] Abir Rahali and Moulay A Akhloufi. 2021. MalBERT: Using transformers for cybersecurity and malicious software detection. *arXiv preprint arXiv:2103.03806* (2021).

[39] Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950* (2023).

[40] Thibault Sellam, Dipanjan Das, and Ankur P Parikh. 2020. BLEURT: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696* (2020).

[41] Nikolaos Serketzis, Vasilios Katos, Christos Ilioudis, Dimitrios Baltatzis, and Georgios Pangalos. 2019. Improving forensic triage efficiency through cyber threat intelligence. *Future Internet* 11, 7 (2019), 162.

[42] Blake E Strom, Andy Applebaum, Doug P Miller, Kathryn C Nickels, Adam G Pennington, and Cody B Thomas. 2018. Mitre att&ck: Design and philosophy. In *Technical report*. The MITRE Corporation.

[43] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems* 27 (2014).

[44] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295* (2024).

[45] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[47] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675* (2019).

[48] Chen Zhong, John Yen, Peng Liu, and Robert F Erbacher. 2018. Learning from experts' experience: toward automated cyber security data triage. *IEEE Systems Journal* 13, 1 (2018), 603–614.