Large Language Models for Automated Characterization of Cybersecurity Vulnerabilities using N-Shot Learning

Ayesha S. Dina, Elijah Needham, Denis Ulybyshev
Department of Computer Science
Florida Polytechnic University
4700 Research Way, Lakeland, FL, 33805
{adina, eneedham, dulybyshev}@floridapoly.edu

Abstract

The US National Vulnerability Database is a public repository of cybersecurity vulnerabilities in software and hardware. This repository is maintained by the National Institute of Standards and Technology (NIST) that developed a Vulnerability Description Ontology framework for characterizing vulnerabilities. Despite advancements in secure software development and vulnerability detection techniques, the number of registered cybersecurity vulnerabilities continues to grow. Characterizing vulnerabilities is essential for selecting effective protection mechanisms to prevent or mitigate cybersecurity vulnerabilities in software and hardware and reduce cyber risks. Manual characterization of vulnerabilities is both time-consuming and costly. While many researchers employ Machine Learning (ML)-based methods to predict characterizations, these methods heavily rely on large amounts of labeled training data. To overcome the challenge of limited labeled data, this paper proposes a solution utilizing three Large Language Models (LLMs) - GPT-4o, Llama-3.1-405B, and Gemini-1.5-flash - to automate the characterization of vulnerabilities across 27 categories, grouped into five noun groups. We use both few-shot and zero-shot learning to prompt the LLMs. Our experimental results show that GPT-4o achieves F1-scores of 80%, 90%, 90%, and 73% in the context, impact-method, attack theater, and logical impact noun groups, respectively, using a few labeled samples. Additionally, Llama achieves an F1-score of 83% in the mitigation noun group.

Introduction

The number of cybersecurity vulnerabilities in software and hardware registered in the public National Vulnerability Database (NVD) continues to grow, despite all the efforts to reduce this number (nvd database). As of 26th of January 2025, there are 240,830 cybersecurity vulnerabilities that are officially registered in NVD, synchronized with the Common Vulnerabilities and Exposures (CVE) database, maintained by the MITRE Corporation since 1999 (MITRE Corporation 2025). Characterizing vulnerabilities with high accuracy is critical for selecting effective mechanisms to mitigate cybersecurity vulnerabilities, prevent software from being exploited, and reduce cyber risks. Furthermore, characterizing vulnerabilities in automated mode is highly desirable since manual characterization is costly and time-consuming. Many researchers rely on security experts to manually characterize vulnerabilities (Okutan et al. 2023). To reduce manual costs, Machine Learning (ML) and Deep Learning (DL) based classifiers have been proposed (Manjunatha et al. 2024; Filus and Domańska 2023). However, these classifiers require a large number of samples to effectively train the model. Labeling the samples is costly.

In this paper, we address two key challenges: automating vulnerability characterization based on the Vulnerability Description Ontology (VDO) and tackling the scarcity of labeled samples. To achieve this, we propose a novel approach to automate the characterization of vulnerabilities across 27 categories, organized into five noun groups defined by the VDO. To the best of our knowledge, our approach is the first to utilize LLMs for characterizing cybersecurity vulnerabilities based on the VDO noun groups. Figure 1 illustrates the system architecture for CVE characterization using LLMs. We design a prompt constructor incorporating VDO noun group definitions and a few labeled samples, which is then provided to LLMs to characterize the unlabeled CVE descriptions. We use few-shot and zero-shot learning to prompt three LLMs: *GPT-4o, Llama-3.1-405B*, and *Gemini-1.5-flash*. Our experimental results are promising: GPT-4o model achieves F1-scores of 80%, 90%, 90%, and 73% in the "Context", "Impact Method", "Attack Theater", and "Logical Impact" noun groups, respectively, using a few labeled samples. Llama-3.1-405B model achieves an F1-score of 83% in the noun group "Mitigation".

The rest of the paper is organized as follows: in 'Related Works' chapter we review the related methodologies and solutions proposed by researchers; 'Core Design' chapter presents the core design of our solution, which is evaluated in the chapter 'Experiments'. Chapter 'Conclusions' discusses our findings and concludes the paper.

Related Works

(Okutan et al. 2023) proposes a methodology for automating vulnerability characterization by utilizing Natural Language Processing (NLP) for pre-processing and Machine Learning (ML) combined with novel Information Theoretical (IT) methods for prediction. The study employs VDO noun groups for characterization but relies on ML classi-

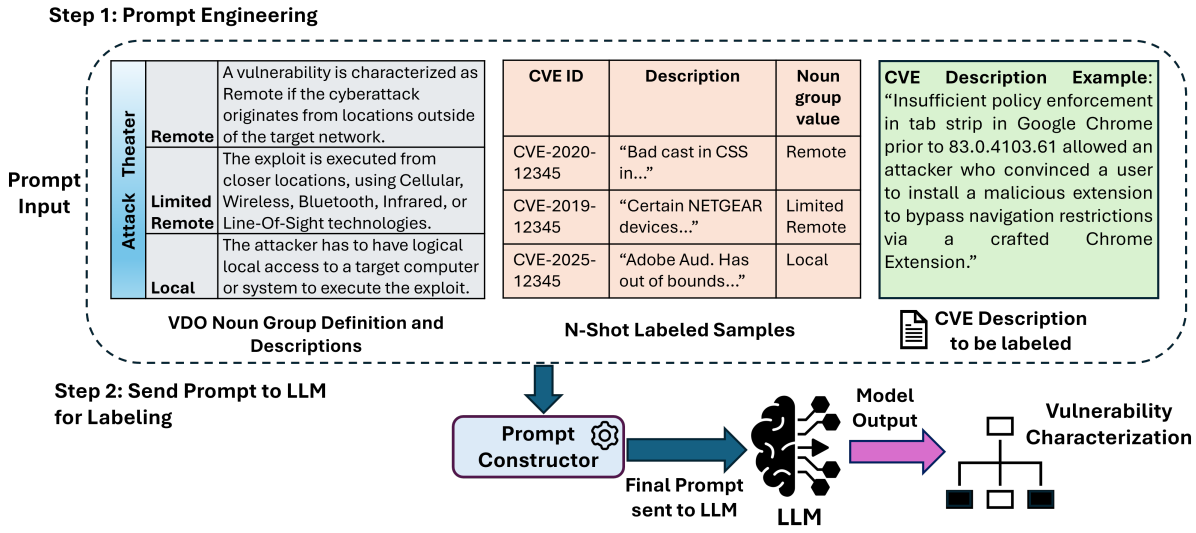

Figure 1: **System Architecture: CVE characterization using LLM with N-shot labeled samples**

fiers, which require a large amount of sample data to train the models effectively. In contrast, our approach works even when there is a scarcity of labeled samples.

There are several methodologies proposed by researchers to categorize software vulnerabilities. Methodology presented in (Yosifova, Tasheva, and Trifonov 2021) aims to predict vulnerability types in the CVE database by applying ML classifiers to analyze CVE records. (Ekle and Ulybyshev 2024) proposed an approach to categorize software vulnerabilities based on their descriptions in NVD. Their solution assigns a label with the vulnerability type, for example, "SQL Injection" or "Buffer Overflow", to the software-related CVEs. Their proposed approach provides high accuracy for categorizing software vulnerabilities, but uses an oversampling technique that may impose space overhead since the size of an input dataset may increase significantly.

(Blinowski and Piotrowski 2020) presented a methodology to classify CVEs in the context of IoT systems. They used the Support Vector Machine (SVM) algorithm on selected subset of CVEs to classify new vulnerability records, with the goal to automatically recognize vulnerable IoT devices. (Kiran et al. 2021) proposed software vulnerability categorization based on CVE records from two years, 2019 - 2020. (Edkrantz and Said 2015) utilized machine learning algorithms to automatically predict unseen vulnerabilities based on prior exploit patterns. As vulnerability data source, they used NVD and the Exploit Database (EDB). (Bozorgi et al. 2010) proposed a ML-based methodology to predict the likelihood and timeframe for individual vulnerabilities being exploited in software. (Wang and Guo 2010) applied Bayesian networks to automate categorization of vulnerabilities based on their types. Our solution relies on LLMs for CVE characterization.

(Zhang et al. 2024) employ LLMs for vulnerability characterization, utilizing discriminative fine-tuning techniques to enhance performance. Our approach automates categorization of both software and hardware vulnerabilities based on the VDO and does not require a large amount of labeled data for training.

Core Design

This work focuses on reducing manual effort in characterizing and describing vulnerabilities by leveraging the NIST VDO framework. To achieve this, we leverage LLMs, including GPT-4o, Llama, and Gemini, for vulnerability characterization. In this section, we discuss the systematic steps for CVE characterization using LLMs.

Vulnerability Description Ontology (VDO) noun group definition

The key components of the NIST framework are depicted in Figure 2, showcasing CVE characterization through vulnerability noun groups (Booth 2016). Each noun group is represented by a box, with the noun groups used in this study highlighted using solid line borders. The headers classify the noun groups as follows: black for mandatory, blue for recommended, and orange for optional, with the corresponding set of applicable values listed within each noun group.

In the cybersecurity community, the following aspects are considered to be important for understanding: (1) the attack avenues through which cyberattacks occur; (2) the vulnerable components of a target system; (3) the methods used in the attack; (4) the potential impacts of successful exploitation; and (5) the applicable mitigation techniques for addressing exploited vulnerabilities (Massacci and Nguyen 2010; Okutan et al. 2023). To address these concerns, this study chooses five noun groups namely *Attack Theater, Context, Impact Method, Logical Impact*, and *Mitigation*.

Prompt Constructor

Prompt engineering is the process of designing and refining prompts to effectively guide LLMs (White et al. 2023).

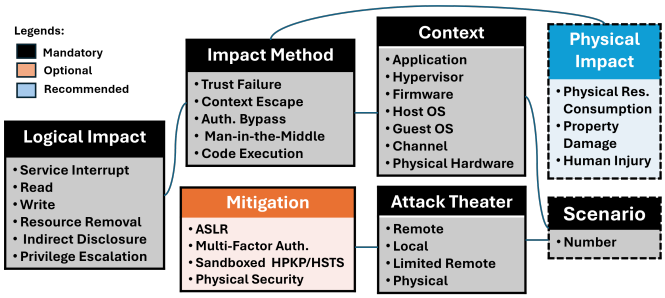

Figure 2: **Partial diagram of Vulnerability Description Ontology (VDO)**

Prompts are instructions provided to an LLM to define rules, automate tasks, and ensure specific qualities and quantities in the generated output. They act as a form of programming, enabling customization of the outputs and interactions with the LLM. For automatic CVE characterization we prepare our Prompt constructor S that can be defined as

$$S = \{V_G, I, N, C\}$$

, where V_G is a set of VDO descriptions of noun group G (e.g., *Attack Theater, Impact Method*). For instance, when we prompt the model to characterize a CVE based on the Attack Theater noun group, the prompt includes only the definition of this noun group along with its corresponding classification labels — Remote, Limited RMT, Local, and Physical — ensuring that only the relevant attributes are considered. I specifies the classification criteria and model output requirements as instructions, N provides N-shot examples, meaning a few labeled samples are included as demonstrations, and C represents the CVE description to be labeled.

Instructions The model is given instructions with two key components. First, it must classify a given CVE description based on the provided VDO noun group definitions and number of examples we give it. Second, we prompt the model to generate its output in a format which can support both single- and multi-labeled characterizations. This standardized output format is necessary because certain CVE descriptions may be associated with multiple labels within a noun group. For example, a CVE that describes a specific vulnerability may contain features corresponding to both the Address Space Layout Randomization *(ASLR)* and *Multi-Factor Authentication* attributes within the *Mitigation* noun group. In other cases, only one of these attributes, or none at all, may be applicable. To ensure consistency in handling both **Single and Multi-Label Classification**, the model is instructed to generate its characterizations using a binary vector representation as follows:

$$O = [X_1, X_2, X_3, \ldots, X_i], \quad X_j \in \{0, 1\}$$

Here, $\mathbf{X_j}$ represents a specific attribute or noun group value within a noun group (e.g *Code Execution* within the *Impact Method* noun group), where (1) indicates the presence of the attribute and (0) its absence. The total number of attributes for a noun group is denoted by **i**, and each index **j** corresponds to a specific attribute in this set. By enforcing this standardized output format, we can efficiently compute performance metrics and assess the models accuracy in classifying CVE descriptions using both single-label and multi-label noun group attributes.

N-Shot Learning N-shot learning is an In-Context-Learning (ICL) paradigm where a model is provided with N labeled examples within a prompt to guide its predictions without requiring fine-tuning or explicit weight updates. This approach is particularly useful in scenarios where labeled data is scarce, allowing LLMs to infer patterns from a limited number of examples. It allows us to examine how the availability of labeled examples influences model performance across different noun groups. Some existing work uses the Few-Shot Learning (FSL) for classification problem in different domain (Agarwal et al. 2024; Ayesha S and AB 2023). We utilize FSL and zero-shot learning. FSL relies on only a small number of labeled samples per class. In our study, we randomly select N labeled samples (N-shot) for each of the A classes. Additionally, we conduct experiments using zero-shot learning, where no labeled samples are used. For each training iteration, we take N samples from each class to produce n-shot setting, where $N \epsilon \{0, 1, 5, 10\}$. In our work, N represents the number of labeled examples provided per attribute within a noun group. Each noun group e.g. *Attack Theater, Context, Impact Method, Logical Impact*, and *Mitigation* contains a set of predefined attributes, and for a given setting of N, the model is supplied with N labeled instances per attribute to aid in classification. When N = 0, the model relies solely on its pre-trained knowledge and the noun group definitions and classification instructions without any labeled examples.

CVE Description The last parameter of our prompt constructor S is the specific CVE description C that we want the LLM to characterize. All CVE entries within the NVD contain information such as the CVE identifier (e.g. CVE-2020-32132), the CVE description, affected products, and impact metrics (MITRE Corporation 2025). However, in this study we only provide the CVE description for the N-shot examples with labels and for the characterization of the CVE. CVE descriptions typically contain relevant information regarding a CVE, such as the specific details of software or hardware vulnerability, affected versions, vendor names, attacker type, and root causes that may have lead to the vulnerability. CVE descriptions vary in detail and consistency, often omitting critical impact information or affected components. These inconsistencies challenge traditional ML models, which rely on structured data and struggle with sparse or ambiguous descriptions. LLMs, however, excel at leveraging context, filling in missing details, and generalizing from limited examples. Unlike conventional models, they can grab the significant keywords from the description, which are related to different noun groups, improving classification results even with minimal training data. In this work, we evaluate LLMs' ability to bridge these gaps, enhancing automated CVE characterization beyond the capabilities of traditional ML approaches.

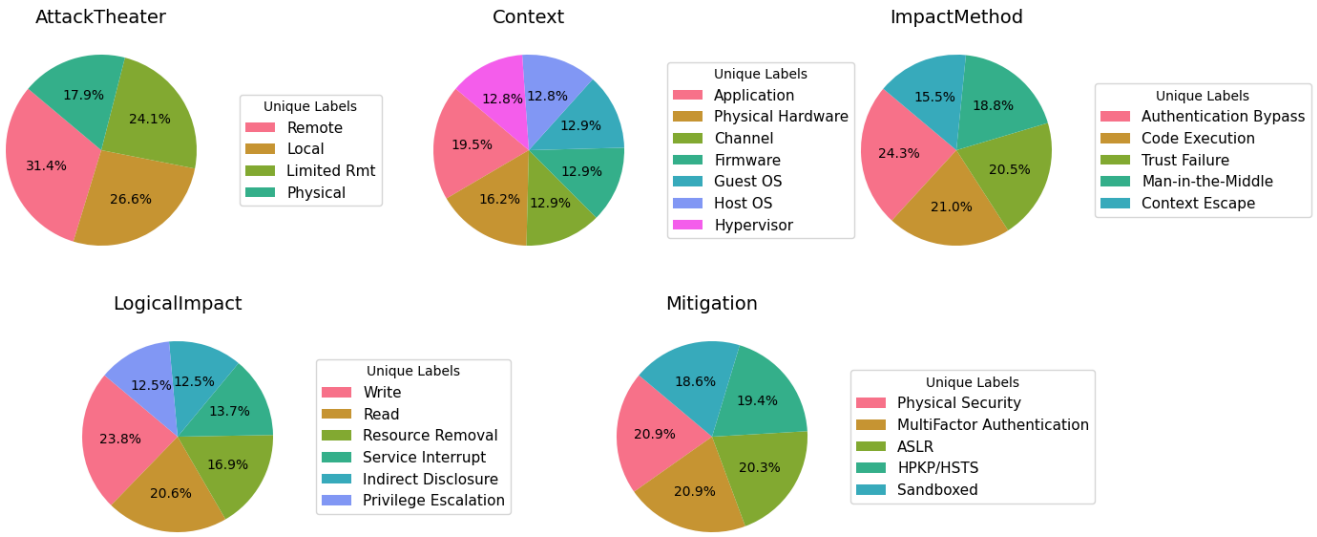

Figure 3: **Data distribution of various noun groups along with their respective noun group values or attributes**

Models

In this study, we leverage LLMs for vulnerability characterization, utilizing three models: GPT-4o, Llama-3.1-405B, and Google Gemini 1.5 Flash. These models exhibit strong generalization capabilities and state-of-the-art performance, having been trained on billions of parameters (Zhang et al. 2024). Their efficiency in processing large volumes of text makes them ideal for the task of vulnerability characterization. This section provides an overview of the LLMs used in our experiments.

GPT-4o is a Transformer-based model from OpenAI (openAI 2022), pre-trained to predict the next token in a document. Its post-training alignment process enhances performance in terms of factual accuracy and adherence to desired behavior (Achiam et al. 2023). We utilize this pre-trained GPT-4o model to characterize vulnerabilities with the prompt constructor S. **Llama-3.1-405B** is developed by Meta AI as part of their Llama series, designed for a variety of NLP tasks (Dubey et al. 2024). This dense Transformer model, with 405 billion parameters, processes information within a context window of up to 128,000 tokens. Its enhanced pre-training and post-training processes ensure high-quality results. We utilize this pre-trained model for vulnerability characterization. **Google Gemini 1.5 Flash** is a transformer decoder model with a 2M+ context and multimodal capabilities, similar to Gemini 1.5 Pro. It is optimized for efficient tensor processing unit (TPU) utilization, offering reduced latency for model serving. (Team et al. 2024).In our study, we employ all three pre-trained LLMs with the same prompt constructor and varying shot settings to characterize cybersecurity vulnerabilities.

Experiments

Experimental details

For our experiments with LLMs, we use OpenAI's API (OpenAI 2024) for GPT-4o and Google Cloud's Vertex AI (gcp vertex AI) for Llama-3.1-405B and Gemini-1.5-flash. The models are not fine-tuned; instead, inferences are performed directly on the dataset. To ensure deterministic outputs and maximize consistency in CVE characterization, the temperature is set to 0, top-p to 1, and the seed parameter is fixed. These settings minimize prediction variability, ensuring greater reliability and reproducibility in automated classification. The primary limitation encountered is rate limiting, particularly for OpenAI, which allows approximately 10,000 requests and 2 million tokens per minute for the GPT-4o model. To address this, execution is structured sequentially, and in instances where rate limits are exceeded, the program is paused for one minute to allow for token replenishment. Similar rate limits are encountered while running Llama and Gemini's models' via Vertex AI, and are addressed in the same manner. Token constraints are well accommodated, as the prompts range between 700 and 3,000 tokens per request, within the supported context windows of each model: GPT-4o supports 128,000 tokens, Llama-3.1-405B supports 128,000 tokens, and Gemini-1.5-flash supports 1,048,576 tokens. (Llama Team 2024) (OpenAI 2024). We apply prompt engineering techniques to optimize efficiency, particularly by placing static components (VDO definitions and instructions) at the beginning of each prompt to leverage caching mechanisms, thereby reducing computational overhead and improving response times.

We evaluate LLM performance using two baseline models, K Nearest Neighbors (KNN) and Naive Bayes (NB), commonly employed for text classification tasks (Sreemathy and Balamurugan 2012). For pre-processing, text is converted to lowercase, numbers and punctuation were removed, and stop-words were filtered out. The text is then tokenized, and term frequency-inverse document frequency (TF-IDF) vectorization is applied to transform it into numerical features. The dataset is split into 60% training and 40% testing using an ordered split to ensure consistent inputs for both models.

Model Name	Shot	Context	ImpactMethod	AttackTheater	LogicalImpact	Mitigation
KNN	-	0.6252	0.7798	0.7938	0.4802	0.5735
NB	-	0.5060	0.8116	0.6619	0.5126	0.5417
GPT	0-shot	0.7291	0.8390	0.8339	0.7142	0.7883
	1-shot	0.7992	0.8628	0.8589	0.7235	0.7940
	5-shot	**0.8015**	0.8863	**0.9104**	0.7573	0.7788
	10-shot	0.8001	**0.9057**	0.9068	0.7301	0.8117
Llama	0-shot	0.5670	0.6885	0.7870	0.6806	0.7480
	1-shot	0.6444	0.8472	0.8366	0.7638	0.8000
	5-shot	0.7073	0.8930	0.9037	0.7912	0.8305
	10-shot	0.7171	0.8898	0.9105	0.7345	**0.8380**
Gemini	0-shot	0.6819	0.7610	0.6539	0.6559	0.5391
	1-shot	0.7179	0.8307	0.8019	0.8010	0.6050
	5-shot	0.7298	0.8386	0.8861	0.8266	0.7036
	10-shot	0.7464	0.8693	0.8881	**0.8313**	0.7645

Table 1: **F1-Scores of models across different noun groups and Few-Shot settings**

Dataset

The dataset, generated by (Okutan et al. 2023), is derived from CVEs (prior to CVE-2020-14000) in the NVD database. They selected CVEs to ensure a balanced distribution across the noun group values within the studied group. To prepare and label the data, five security researchers with 2–15 years of experience spent over 3,000 person-hours creating training datasets for the five VDO noun groups. All reports are peer-reviewed to minimize biases, and any uncertainties are collectively discussed to maintain consistency. The process involves five steps: (1) Subject Matter Experts (SMEs)' understanding of the noun groups, (2) labeling and annotation sessions, (3) self-reported confidence, (4) peer discussion sessions, and (5) peer review and validation. We use this dataset to evaluate our proposed solution. Figure 3 illustrates data distribution of various noun groups along with their respective noun group values or attributes.

Evaluation Metrics

To assess model performance, this study employs multiple evaluation metrics, including Precision, Recall, and F1-score to evaluate the models performance across different noun groups. Predictions fall into four categories: True Positive (TP), where a CVE is correctly classified under its true VDO label; False Positive (FP), where a CVE is incorrectly classified under a label it does not belong to; True Negative (TN), where a CVE is correctly classified as not belonging to a label; and False Negative (FN), where a CVE belongs to a label but is misclassified under another. Precision measures the fraction of correctly predicted labels out of all predicted labels for a given class. Recall measures the fraction of correctly predicted labels out of all actual occurrences of that label. The F1-score is the harmonic mean of Precision and Recall, balancing both metrics:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

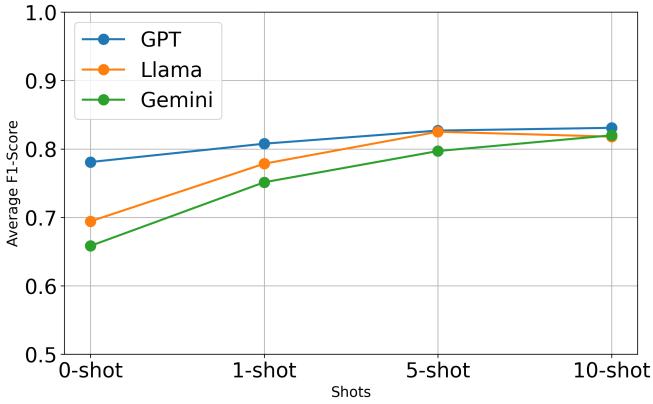

Figure 4: **Comparison of average F1-scores of LLMs for varying N-shot values across the noun groups**

Results

In this section, we present the evaluation results of LLMs and ML classifiers for automated cybersecurity vulnerability characterization. We compare GPT-4o, Llama 3.1 405B, and Gemini 1.5-flash with traditional models KNN and NB across five noun groups. The effectiveness of the models is assessed using the macro F1-score across various classification settings. The performance of the models on F1-scores are provided in Table 1.

Context. In the "Context" category, GPT consistently achieves the highest F1-scores, peaking at 0.8015 in the 5-shot setting. Llama shows notable improvement with increased shots, rising from 0.5670 in the 0-shot setting to 0.7171 at 10-shot. Gemini follows a similar trend, progressing from 0.6819 in 0-shot to 0.7464 at 10-shot. KNN and NB both perform quite poorly in the context category, with scores of 0.6252 and 0.5060 respectively. Even in the 0-shot setting, GPT and Gemini outperformed the baseline models by up to 16%. In the 5-shot setting, GPT improved by over 28%. These results highlight the ability of LLMs to effectively leverage few-shot examples for context classification.

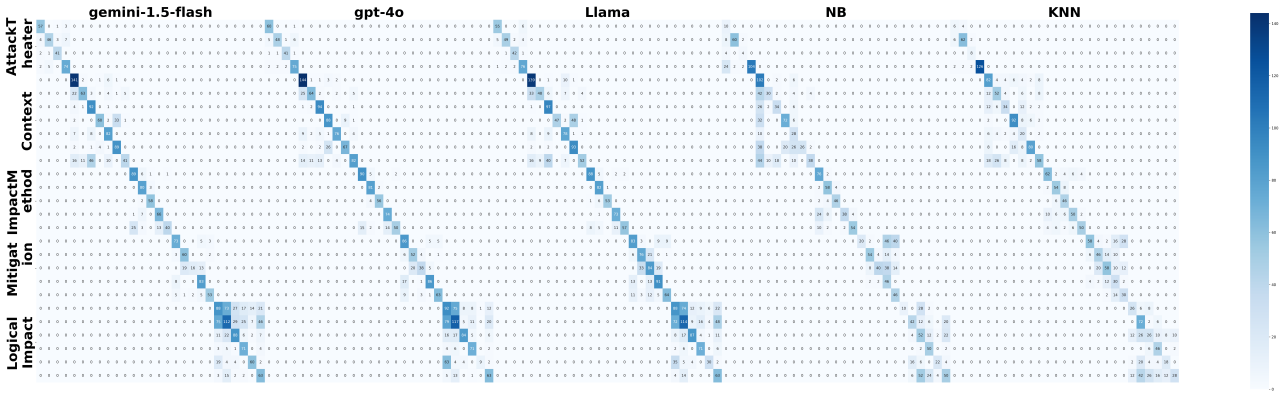

Figure 5: **Confusion matrices for all noun groups on various models**

Impact Method. For "Impact Method" classification, GPT achieves the highest score of 0.9057 at 10-shot, followed closely by Llama at 0.8930 in 5-shot and 0.8898 in 10-shot. Gemini also demonstrated strong performance, reaching 0.8693 at 10-shot. Among the baseline models, KNN and NB both perform quite better than they did on other noun groups with scores of 0.7798 and 0.8116, outperforming Gemini and Llama when using 0-shot methods. Even still, GPT was able to outperform KNN **by 8%** in the 0-shot setting. On 1-shot setting and few-shot setting, all LLMs outperform the baseline models **by 6–16%**.

Attack Theater. The "Attack Theater" category follow a similar pattern, with GPT reaching its highest F1-score of 0.9104 at 5-shot, while Llama slightly surpassed it with a peak of 0.9105 at 10-shot. Gemini also improved significantly from 0 to 10-shot with an F1-score increase **by over 36%** to an F1-score of 0.8898. For this category, KNN (0.7938) outperforms NB (0.6619) but still remains below GPT's 0-shot performance of 0.8339. KNN is comparable to Llama's 0-shot result (0.7870), indicating that classical ML models can be a little more competitive when LLMs have not seen any examples. Once few-shot examples are introduced however, all three LLMs surpass KNN and NB, **with up to 15% higher** F1-scores in the "Attack Theater" noun group.

Logical Impact. In the "Logical Impact" category, both Gemini and Llama models perform very well as the n-shot number increases, with Gemini peaking at .8313 in the 10-shot setting. In the 0-shot setting GPT performs slightly better, but was surpassed by Gemini and Llama for all other n-shot values. The baseline models, KNN and NB both perform very poorly in this category with scores of 0.4802 and 0.5126 respectively. By contrast, even the lowest performing LLM at 0-shot result (Gemini at 0.6559) outperforms them both **by 28%**, and **over 62%** when comparing them to the best performing LLMs in the "Logical Impact" category.

Mitigation. Finally, in the "Mitigation" category, Llama model achieves the highest F1-score of 0.8380 at 10-shot, narrowly surpassing GPT's 0.8117 in the same setting. Gemini consistently increases its F1-score by 7–10% for the increase in the N-shot value; however, its peak F1-score (0.7645 at 10-shot) was still the lowest of the three LLMs in this category. The baseline models also performed quite poorly in this category as well, with scores of 0.5735 and 0.5417 for KNN and NB. Both GPT and Llama greatly outperform these baseline models, even in the 0-shot setting, **with F1-scores up to 45% higher.**

Figure 4 presents the comparison of the LLMs' average performance across all noun groups. GPT-4o consistently achieves the highest average F1-score across the five noun groups under different N-shot settings, outperforming both Llama and Gemini. Llama and Gemini perform poorly in the zero-shot setting; however, their performance improves as the number of shots increases, achieving an average F1-score above 0.8. Figure 5 presents the confusion matrices for all noun groups across baseline models and LLMs. The right side of this figure displays the confusion matrices for KNN and NB classifiers. As shown, both baseline models misclassify many noun group values across various noun groups. Conversely, LLMs demonstrate better performance in classifying noun group values.

Conclusions

This paper addresses two challenges: *(i)* automating vulnerability characterization using the Vulnerability Description Ontology (VDO) and *(ii)* the scarcity of labeled samples. We propose a novel approach using LLMs for characterizing cybersecurity vulnerabilities across 27 categories within five VDO noun groups. A prompt constructor incorporating VDO definitions and a few labeled samples is used to guide LLMs in characterizing unlabeled CVE descriptions. Using few-shot and zero-shot learning, we evaluate GPT-4o, Llama-3.1-405B, and Gemini-1.5-flash models. GPT-4o achieves F1-scores of 80%, 90%, 90%, and 73% in the "Context", "Impact Method", "Attack Theater", and "Logical Impact" noun groups, respectively, while Llama-3.1-405B achieves 83% in "Mitigation". Compared to baseline models trained in a fully supervised manner, our approach achieves competitive results using significantly fewer labeled samples. As a future work, we plan to build our own AI model for better VDO characterization aiming to characterize previously unseen vulnerability descriptions that might be inconsistent, unstructured, informal, abbreviated, or contain jargon-heavy text.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Agarwal, R.; Singh, A.; Zhang, L. M.; Bohnet, B.; Rosias, L.; Chan, S.; Zhang, B.; Anand, A.; Abbas, Z.; Nova, A.; et al. 2024. Many-shot in-context learning. *arXiv preprint arXiv:2404.11018*.

Ayesha S, D., and AB, S. 2023. Fs3: Few-shot and self-supervised framework for efficient intrusion detection in internet of things networks. In *Proceedings of the 39th Annual Computer Security Applications Conference*, 138–149.

Blinowski, G. J., and Piotrowski, P. 2020. Cve based classification of vulnerable iot systems. In Zamojski, W.; Mazurkiewicz, J.; Sugier, J.; Walkowiak, T.; and Kacprzyk, J., eds., *Theory and Applications of Dependable Computer Systems*, 82–93. Cham: Springer International Publishing.

Booth, H. 2016. Draft nistir 8138, vulnerability description ontology (vdo). *National Institute of Standards and Technology (NIST), Tech. Rep*.

Bozorgi, M.; Saul, L. K.; Savage, S.; and Voelker, G. M. 2010. Beyond heuristics: learning to classify vulnerabilities and predict exploits. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, 105–114. New York, NY, USA: Association for Computing Machinery.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Edkrantz, M., and Said, A. 2015. Predicting cyber vulnerability exploits with machine learning. In *Thirteenth Scandinavian Conference on Artificial Intelligence*, 48–57. IOS Press.

Ekle, O. A., and Ulybyshev, D. 2024. Enhanced categorization of cybersecurity vulnerabilities. In *2024 IEEE 15th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 800–806.

Filus, K., and Domańska, J. 2023. Software vulnerabilities in tensorflow-based deep learning applications. *Computers & Security* 124:102948.

gcp vertex AI. Vertex ai. `https://cloud.google.com/vertex-ai?hl=en`.

Kiran, S.; Rajper, S.; Shaikh, R. A.; Shah, I. A.; and Danwar, S. H. 2021. Categorization of cve based on vulnerability software by using machine learning techniques. *International Journal (Toronto, Ont.)* 10(3).

Llama Team, A. 2024. The llama 3 herd of models. *Meta AI Research*. Available online: `https://llama.meta.com/`.

Manjunatha, A.; Kota, K.; Babu, A. S.; et al. 2024. Cve severity prediction from vulnerability description-a deep learning approach. *Procedia Computer Science* 235:3105–3117.

Massacci, F., and Nguyen, V. H. 2010. Which is the right source for vulnerability studies? an empirical analysis on mozilla firefox. In *Proceedings of the 6th international workshop on security measurements and metrics*, 1–8.

MITRE Corporation. 2025. CVE Program Overview. Accessed: January 26, 2025.

nvd database. National vulnerability database. `https://nvd.nist.gov/`. Accessed: 2025-01-22.

Okutan, A.; Mell, P.; Mirakhorli, M.; Khokhlov, I.; Santos, J. C.; Gonzalez, D.; and Simmons, S. 2023. Empirical validation of automated vulnerability curation and characterization. *IEEE Transactions on Software Engineering* 49(5):3241–3260.

openAI. 2022. Introducing chat gpt. `https://openai.com/blog/chatgpt,`.

OpenAI. 2024. Gpt-4o system card. Technical report, OpenAI. Available online: `https://openai.com/gpt-4o-system-card`.

Sreemathy, J., and Balamurugan, P. 2012. An efficient text classification using knn and naive bayesian. *International Journal on Computer Science and Engineering* 4(3):392.

Team, G.; Georgiev, P.; Lei, V. I.; Burnell, R.; Bai, L.; Gulati, A.; Tanzer, G.; Vincent, D.; Pan, Z.; Wang, S.; et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Wang, J. A., and Guo, M. 2010. Vulnerability categorization using bayesian networks. In *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research*, CSIIRW '10. New York, NY, USA: Association for Computing Machinery.

White, J.; Fu, Q.; Hays, S.; Sandborn, M.; Olea, C.; Gilbert, H.; Elnashar, A.; Spencer-Smith, J.; and Schmidt, D. C. 2023. A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382*.

Yosifova, V.; Tasheva, A.; and Trifonov, R. 2021. Predicting vulnerability type in common vulnerabilities and exposures (cve) database with machine learning classifiers. In *2021 12th National Conference with International Participation (ELECTRONICA)*, 1–6.

Zhang, J.; Wang, C.; Li, A.; Sun, W.; Zhang, C.; Ma, W.; and Liu, Y. 2024. An empirical study of automated vulnerability localization with large language models. *arXiv preprint arXiv:2404.00287*.