Classifying IoT-Related Vulnerabilities in the CVE Dataset Using Large Language Models

TIMON BELD, University of Twente, The Netherlands

The Internet of Things (IoT) continues to expand rapidly. Meanwhile, vulnerabilities in IoT systems have become increasingly prevalent. Despite this, the Common Vulnerabilities and Exposures (CVE) database, a key resource in cybersecurity, does not explicitly indicate whether a vulnerability is related to IoT. Previous studies primarily rely on keyword-based searches and manual review to identify IoT-related vulnerabilities, which is time-consuming and may fail to capture more nuanced cases. In this paper, we assess the effectiveness of large language models (LLMs) in classifying CVEs as IoT-related or not and compare their performance against fine-tuned Natural Language Processing (NLP) based classifiers. We then investigate which specific components of a CVE entry, such as the description or affected items, contribute most significantly to the classification decision, providing insights into the internal decision-making process of the LLMs. Our goal is to go beyond simple keyword detection, reduce manual work, and expand the current datasets of IoT vulnerabilities. Results show that LLMs have high potential in automatically classifying CVEs with an accuracy of 92% without any fine-tuning or the need for labeled training data. It significantly reduces the manual effort required for processing such large vulnerability datasets and offers researchers and security professionals a powerful tool for enhancing IoT security.

Additional Key Words and Phrases: IoT vulnerabilities, CVE, large language models, dataset labeling

1 INTRODUCTION

The Internet of Things (IoT) is expanding at an extraordinary rate, with forecasts suggesting that the number of connected devices could reach 75.44 billion by 2025 and that the economic impact of IoT technologies ranges between $2.7 trillion and $6.2 trillion [21]. Despite this rapid growth, security remains a major concern. For example, the smart home sector alone was expected to surpass $40 billion by 2020, yet there are still no universal security standards for IoT devices. As a result, many products remain vulnerable to attacks targeting their physical components, networks and encryption mechanisms [8].

A notable example of these risks is the Mirai botnet, which exploited default credentials in IoT devices to create a large network of compromised systems capable of launching devastating Distributed Denial-of-Service (DDoS) attacks [23]. Mirai infects vulnerable IoT devices, mainly targeting products such as IP cameras and home routers. It was used in major DDoS attacks, including the disruption of a DNS provider and several high-profile websites [26].

Therefore, given the high stakes involved, it is crucial to identify vulnerabilities in IoT systems. Prior research has shown that IoT vulnerabilities often stem from common, recurring issues [18] and that auditing frameworks typically assess whether a device can be exploited based on previously disclosed vulnerabilities. However, despite the existence of extensive vulnerability datasets such as the NVD CVE database, IoT-specific vulnerabilities are not explicitly labeled. Prior efforts to identify IoT-related CVEs have relied on keyword-based searches and manual review. However, this approach is time-consuming and not scalable, limiting its effectiveness in systematically identifying IoT vulnerabilities.

These limitations highlight the need for an effective and efficient method to classify vulnerabilities in large CVE datasets. Recent progress in large language models has shown impressive performance across various natural language processing tasks, particularly in text classification, often without the need for labeled data or extensive fine-tuning. For instance, Kostina et al. [14] demonstrated that LLMs like LLaMA3 and GPT-4 can outperform traditional classifiers on complex multiclass and binary tasks, simply by applying effective prompts. Similarly, Sun et al. [20] demonstrated that few-shot prompting techniques can achieve near-supervised performance using as few as 16 examples per class. Despite these advances, whether LLMs can accurately label CVEs as IoT-related remains unexplored.

Therefore, in this work, we leverage large language models to automatically classify CVEs as either IoT-related or not IoT-related. We evaluate a diverse set of LLMs from different model families and parameter sizes in a zero-shot setting, using a carefully constructed and manually validated dataset. To assess how effective LLMs are compared to other available technologies, we also fine-tune several state-of-the-art NLP-based classifiers on the same task.

In addition to measuring standard performance metrics, we analyze feature importance to determine which input fields contribute most significantly to LLM classification decisions. This leads us to the following research questions (RQ) as the basis of our research:

- **RQ1:** How do different LLMs compare in performance on the task of IoT vulnerability classification?
- **RQ2:** How effectively can large language models classify CVE entries as IoT-related compared to fine-tuned classifiers?
- **RQ3:** Which fields or features of a CVE entry (e.g. description, affected items, published date) contribute most significantly to determining IoT relevance?

The rest of the paper is organized as follows. In Section 2, we review prior studies. Section 3 explains our methodology and the selected LLM models. Section 4 elaborates on the dataset, the conducted analysis, and the results. In Section 5, we present the discussion, and we conclude in Section 6. All supporting materials used in this study can be found in the references section, as well as in the appendices. The final datasets, along with all intermediate results, are publicly available on the GitHub repository at [4].

2 RELATED WORK

In this section, we review two main strands of related work. First, we discuss studies that have investigated IoT vulnerabilities at scale and highlighted the complexity of reliably identifying and classifying IoT-related security issues. Second, we review recent research that has applied large language models to various text classification tasks, demonstrating their potential for understanding and categorizing unstructured textual data.

TScIT 43, July 4, 2025, Enschede, The Netherlands

2.1 IoT Vulnerability Analysis

We base our study on the labeled dataset created by Chen et al. [7], who manually classified CVEs as IoT-related or non-IoT-related. Their empirical study collected 198,464 CVE entries from the National Vulnerability Database (NVD) and applied an IoT-specific keyword filtering and manual verification process to construct a dataset containing 1,739 IoT-related and 196,725 non-IoT-related CVEs.

Zhao et al. [27] performed a large-scale empirical analysis of vulnerabilities introduced by third-party components in IoT firmware. By analyzing 34,136 firmware images, they identified 584 TPCs and 429 CVEs. They found that outdated components are major sources of vulnerabilities. Their work also shows that a single CVE can propagate across many devices due to shared components and libraries.

Hulayyil and Li [13] proposed an IoT vulnerability detection and mitigation system that does not rely on predefined features. Instead, it processes raw network traffic using a lightweight neural network and a language model. The system runs in real time and achieves high accuracy in detecting and responding to threats during live operation.

2.2 LLMs for Text Classification

Abburi et al. [1] proposed an ensemble-based method for text classification, combining the outputs of multiple large language models as features for a traditional fine-tuned machine learning classifier. Their findings showed strong performance in both binary and multi-class tasks across English and Spanish datasets.

Vajjala and Shimangaud [24] compared zero-shot, few-shot, and synthetic-data approaches across 32 datasets in eight languages. While zero-shot works well for sentiment tasks, few-shot tuning and synthetic data often perform better in more complex cases, especially across different languages.

Tony et al. [22] introduced LLMSecEval, a benchmark of 150 natural-language prompts targeting MITRE's Top 25 CWEs for evaluating code-security capabilities of LLMs. Each prompt includes secure implementation examples, enabling assessment of vulnerability generation potential in model outputs.

3 METHODOLOGY

In order to identify IoT-related CVEs, we first propose a precise definition of the IoT ecosystem and IoT-related CVEs. We then select several LLMs publicly available through the Ollama framework [19], covering various LLM sizes and families. Each model is then prompted via a structured input that includes our definition of an IoT ecosystem followed by the CVE description. The performance of LLMs is evaluated on a ground-truth dataset comprising IoT and non-IoT CVEs. We further compare LLMs against several state-of-the-art NLP classifiers in terms of performance and execution time.

3.1 IoT ecosystem definition

In this study, we define an IoT system as comprising three primary components: device, network, and application [11, 15]. The device component refers to physical objects embedded with sensors, software, and connectivity features that allow it to collect, exchange, and act on data over the internet or other networks without requiring direct human intervention. Examples include—but are not limited to—routers, switches, IP cameras, smart home devices, etc. Moreover, components that enable IoT connectivity, such as embedded modules, smart controllers, and network interfaces are considered part of the IoT devices [12]. The network component serves as the communication layer that facilitates data transfer and connectivity between devices. This includes, but is not limited to, technologies and protocols such as BLE, CoAP and MQTT. The application component is responsible for processing data into actionable insights and delivering user-facing functionality. Examples include companion mobile apps, user interfaces, and platforms like IFTTT and SmartThings.

Given the above definition of an IoT system, we define an IoT-related CVE as one that impacts or targets essential components of an IoT system, including devices, networks, and applications, and is primarily or exclusively deployed within IoT contexts. It is worth noting that we do not consider the smartphone, computer, or tablet as an IoT device.

3.2 Chosen Large Language Models

We selected six LLMs based on factors such as model size, architecture, instruction-following capability, and efficiency on local hardware. Below is an overview of the selected models and a brief explanation of why each was chosen:

- **LLaMA 3.1 (8B):** We selected *LLaMA 3.1 (8B)* primarily because it is known for its strong performance in processing complex, technical text, which often contains dense terminology. It is developed by Meta and one of the most widely adopted open-weight models in the industry, making it a strong benchmark for evaluating classification effectiveness. It has 8B parameters and a disk size of 4.7GB [16].
- **Mistral (7B):** *Mistral (7B)* was selected for its compact architecture and strong reasoning capabilities, making it well-suited for local inference without compromising classification performance. On paper, it offers performance comparable to *LLaMA 3.1* while requiring fewer resources. We chose the 7B parameter version, which has a disk size of 4.1GB, making it a lightweight yet powerful model for our use case [3].
- **Phi 4 Mini (3.8B):** *Phi-4 Mini (3.8B)* is a compact and efficient language model released by Microsoft in February 2025 as part of the Phi-4 family. It's the newest model in our evaluation. We selected the 3.8B parameter variant to explore the impact of model size on classification accuracy, particularly in handling complex and nuanced CVE descriptions. With a disk size of 2.5GB, it was the smallest model in our comparison, allowing us to assess whether a higher parameter count and smaller model results in a less precise identification [17].
- **DeepSeek-R1 (7B):** *DeepSeek-R1 (7B)* is a 7-billion parameter model developed by DeepSeek AI. With a disk size of 4.7GB, *DeepSeek-R1 (7B)* was selected not only for its technical strengths but also due to its recent rise in popularity within the open-source community. Including it in our evaluation allows us to explore how newly emerging open models perform in comparison to more established baselines like *LLaMA 3.1 (8B)* and *Mistral (7B)* [2].
- **Gemma 3 (4B):** *Gemma (4B)* is a lightweight, instruction-tuned open-weight model developed by Google, with 4 billion parameters and a disk size of approximately 3.3GB. It was selected to provide a useful point of comparison with *Phi-4 Mini (3.8B)* to establish a stronger baseline for small-model effectiveness in CVE classification tasks [9].

- **Gemma 3 (12B):** *Gemma (12B)* is the larger variant in Google's Gemma model family, containing 12 billion parameters and a disk size of approximately 8.1GB. It was selected to examine how models of the same family scale in performance by comparing this larger variant with its smaller 4B counterpart [9].

3.3 Consistent Classification

To ensure consistency in classification and eliminate ambiguity in interpretation, we design the prompt by first providing the exact definition of IoT-related vulnerabilities given in Section 3.1. This is then followed by the inclusion of the following structured fields:

- **affected_items**: A structured list indicating which products, vendors, or systems are affected by the vulnerability.
- **description**: A summary describing the vulnerability, its impact, and technical details.
- **provider_short_name**: Name or abbreviation of the vulnerability data provider or source.
- **published_date**: Date on which the CVE entry was officially published.
- **tags**: Additional labels or metadata assigned to the CVE entry.

This prompt structure ensures that the models receive both the contextual and factual information necessary for consistent decision-making. By clearly stating the definition upfront, we minimize subjective variation and ensure that both LLMs and human annotators apply the same criteria.

3.4 Classification Models

We selected three well-established transformer-based models to serve as the core classifiers in our evaluation pipeline. Each model was chosen based on its strengths in handling text classification tasks and its alignment with the language and structure of CVE descriptions:

- **SciBERT (allenai/scibert_scivocab_uncased):** Trained on a large corpus of scientific and technical text, SciBERT is particularly well-suited to the domain-specific language used in CVE entries. Its familiarity with structured, formal writing makes it ideal for accurately parsing cybersecurity-related content.
- **DistilBERT (distilbert-base-uncased):** This distilled version of BERT offers a lightweight alternative that retains much of BERT's representational power while significantly improving inference speed and efficiency. It is particularly useful for real-time or resource-constrained applications.
- **DeBERTa v3 Small (microsoft/deberta-v3-small):** DeBERTa introduces advanced attention mechanisms and improved positional encoding, resulting in state-of-the-art performance on classification benchmarks. It serves as a high-performing benchmark model in our analysis.

4 ANALYSIS AND RESULTS

In this section, we first explain our system setup, evaluation metrics and dataset. We then present our results and address the research questions.

4.1 System setup

All experiments were conducted on a local workstation equipped with an NVIDIA RTX 2060 GPU featuring 16 GB of dedicated VRAM, paired with 32 GB of system RAM and a multi-core CPU.

This hardware configuration provided sufficient computational resources for evaluating medium-sized LLMs while maintaining reasonable processing times. However, the hardware limitations prevented us from exploring larger-scale models, which typically require significantly more memory and computational power. The full hardware setup and detailed specifications are provided in Appendix A.

4.2 Evaluation Metrics

The following metrics are collected for each model to evaluate classification performance in relation to RQ1 and RQ2:

- **Accuracy**: The overall proportion of correctly classified CVEs.
$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$
- **Precision**: The proportion of CVEs predicted as IoT-related that are truly IoT-related.
$$\text{Precision} = \frac{TP}{TP + FP}$$
- **Recall**: The proportion of actual IoT-related CVEs that are correctly identified.
$$\text{Recall} = \frac{TP}{TP + FN}$$
- **F1 Score**: The harmonic mean of precision and recall, providing a balanced metric when considering both false positives and false negatives.
$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$
- **Execution time (s)**: The average inference time per CVE sample, measured in seconds.
$$\text{Execution Time} = \frac{\text{Total Time}}{\text{Number of Samples}}$$

4.3 Dataset

We utilize the IoT vulnerability classification dataset constructed by Chen et al. [7] as a basis dataset. Their work collected 198,464 CVE entries from the National Vulnerability Database (NVD), covering the period from August 1999 to April 2023. To identify IoT-related vulnerabilities, they created an IoT-specific keyword list derived from the titles and keywords of papers published in top security conferences. This keyword list was then used to filter the CVE database, resulting in a dataset containing 1,739 IoT-related CVEs and 196,725 non-IoT CVEs.

Because the original dataset contains significantly more non-IoT-related CVEs than IoT-related ones, we balanced the data by randomly selecting an equal number of non-IoT CVEs from the same time period as the IoT-labeled entries. This results in a balanced dataset with a 50/50 distribution between IoT and non-IoT CVEs (both consisting of 1,739 CVE's), ensuring that the classification task is not biased by class imbalance.

Afterwards, we divide this dataset into training and testing sets. The training set is used to fine-tune the classifiers, while the testing set is used to evaluate the performance of both the classifiers and LLMs. Specifically, we randomly select 70% of the dataset (2,478 samples) for training, ensuring a balanced 50/50

Table 1. Classification performance for different LLMs on the test set.

Model	Accuracy	Precision	Recall	F1 Score	Avg time per sample (s)
Gemma (12B)	**87.4%**	0.883	0.869	0.876	8.5
LLaMA 3.1 (8B)	85.6%	**0.925**	0.786	0.850	1.7
Mistral (7B)	86.3%	0.879	0.877	**0.878**	1.5
DeepSeek (7B)	84.5%	0.851	0.838	0.845	15.7
Gemma 3 (4B)	83.2%	0.823	0.861	0.842	**1.4**
Phi-4 Mini (3.8B)	77.8%	0.720	**0.898**	0.799	2.2

Table 2. Classification performance for fine-tuned Transformers on the test set.

Model	Accuracy	Precision	Recall	F1 Score	Avg time per sample (s)
SciBERT (scivocab-uncased)	**97.5%**	**0.993**	0.975	**0.984**	0.008
DeBERTa v3 Small	96.0%	0.947	**0.994**	0.969	**0.005**
DistilBERT (base-uncased)	96.2%	0.987	0.961	0.974	0.007

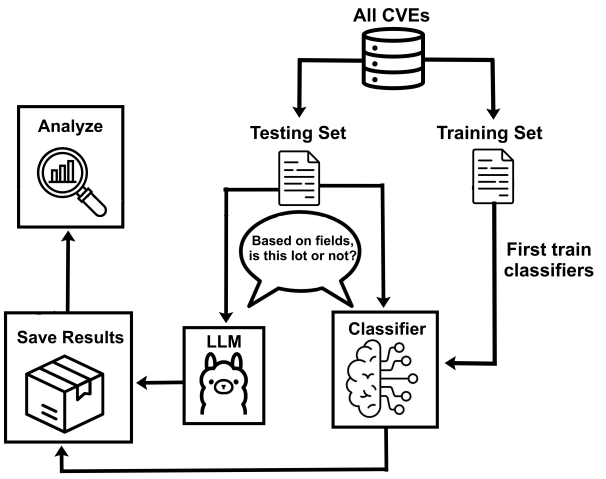

Fig. 1. Evaluation process overview.

split between IoT and non-IoT samples. The remaining 30% (1000 samples) is used for testing. During the fine-tuning process, the training set is further split into training and validation subsets using an industry-standard 80/20 ratio.

We evaluated all selected LLMs using the prompt design described in Section 3.3. Both the LLMs and fine-tuned classifiers were tested on the test set of 1,000 CVEs. and assessed using the evaluation metrics defined in Section 4.2. This ensures a fair and unbiased comparison between the classifiers and the LLMs. For a high level overview of the process, see Figure 1.

4.4 LLM Comparison

In order to answer RQ1, we evaluate the performance of selected LLMs on the test set. Table 1 shows the comparison results. *Mistral (7B)* and *Gemma (12B)* stand out as the top-performing models among all evaluated LLMs. *Gemma (12B)* achieved the highest accuracy (87.4%) and a strong F1 score (0.876), indicating robust overall performance. *Mistral (7B)*, however, delivered a slightly higher F1 score (0.878) despite a bit lower accuracy (86.3%), making its classification quality nearly indistinguishable from that of *Gemma (12B)*. Both models demonstrate a well-balanced trade-off between precision and recall, enabling effective detection of IoT-related CVEs while minimizing false positives and false negatives.

However, it is important to consider the cost of this performance. At 12 billion parameters, *Gemma (12B)* is significantly larger than *Mistral (7B)*, resulting in a higher computational overhead and a 5.7x longer evaluation time per sample.

Notably, *LLaMA 3.1 (8B)* achieved the highest precision (0.925), indicating strong capability in correctly identifying non-IoT CVEs and reducing false positives. However, this came at the cost of a lower recall (0.786), indicating that it misses a larger proportion of true IoT-related cases compared to other models.

Next in performance, *DeepSeek-R1 (7B)* and *Gemma (4B)* demonstrated comparable performance, with F1 scores of 84.5% and 84.2%, respectively. Notably, *Gemma (4B)*, despite being approximately half the size of *DeepSeek-R1 (7B)*, achieved nearly identical performance, while being almost 11× faster. This suggests that even smaller-scale models can maintain strong classification performance and that results may also reflect the strengths of specific model families, such as Gemma, which appears to scale effectively across different parameter sizes.

Phi-4 Mini (3.8B) delivered the weakest results overall, with an accuracy of just 77.8%. It achieved relatively high recall (0.898), but at the expense of precision (0.720), leading to frequent misclassification of non-IoT CVEs and only moderately correct classification of IoT-related ones.

We can clearly see that execution time generally increases with model size. *Gemma (12B)* requires 8.5 seconds per sample, while *DeepSeek-R1 (7B)*, although it performs reasonably well in terms of accuracy, has an average inference time of 15.72 seconds per sample, making it the slowest model by far. This is because DeepSeek attempts to provide an explanation for its decisions and is not well suited for returning a structured response. In contrast, smaller models such as *Mistral (7B)* and *Gemma (4B)* complete inference significantly faster. This trend indicates that execution time is strongly correlated with model size, particularly the number of parameters.

Overall, *Gemma (12B)* demonstrates that scale can lead to meaningful gains in classification performance, while also highlighting the need to consider computational cost. The consistency and margin of improvement suggest that large, general-purpose models can be highly effective, even without task-specific fine-tuning.

We speculate that even larger models with greater parameter capacity may yield further performance gains, pointing to a positive correlation between model size and classification effectiveness.

4.5 Dataset review

The observed classification performance of the LLMs was lower than expected, which raised questions about the validity of the dataset. As a result, we manually inspected a sample of the data to assess its quality. Upon review, we discovered that some entries were incorrectly labeled, certain CVEs marked as IoT-related were not actually IoT-related, and vice versa. For example, the original dataset reports CVE-2017-7050 as IoT-related, while it is related to macOS, which runs on desktop devices and is not IoT-related. Conversely, some entries labeled as non-IoT-related are in fact associated with IoT systems. For instance, CVE-2008-1258 involves a cross-site scripting vulnerability in the web interface of a router. Exploiting this vulnerability could allow an attacker to inject malicious scripts into the router's management panel, potentially hijacking sessions or redirecting traffic from connected IoT devices. This is classified under our definition as IoT-related since it targets a network-connected device that serves as a communication interface between IoT components, affecting both the device and network layers of the IoT architecture. Therefore, relying purely on the original labels may introduce inaccuracies into the dataset.

Therefore, we decided to manually review some CVEs and verify their label. To do so, we utilize the three best performing LLMs from RQ1 to conduct an initial filtering of the test set. Namely these LLMs are:

- **LLaMA 3.1 (8B)**
- **Mistral (7B)**
- **Gemma (12B)**

An overview of the manual review process is provided in Figure 2. For each CVE in the test set, we collect the labels predicted by the three LLMs and apply majority voting. If the resulting consensus label differs from the original label in the dataset, the CVE is flagged for manual inspection. This filtering process resulted in 126 CVEs being selected for manual inspection.

After manually reviewing the marked CVEs, we identified 86 CVEs that were incorrectly labeled in the original dataset. For each of these cases, we reassigned the correct label: CVEs that were incorrectly marked as IoT-related were relabeled as non-IoT, and vice versa. After updating these entries, we integrated the corrected labels back into the full dataset. Since a larger proportion of the misclassifications were originally labeled as IoT-related, the correction process led to a class imbalance. To restore a balanced dataset, we randomly downsampled the non-IoT class until both classes contained an equal number of samples. This resulted in a final dataset consisting of 916 CVEs, with an even 50/50 split between IoT and non-IoT vulnerabilities, which we use as the corrected dataset for the other two research questions. The final validated dataset, along with the original version prior to manual review, is publicly available at [4].

After revising the labels, we evaluated the performance of LLMs on the corrected dataset. Table 3 shows the performance of LLMs on the corrected dataset. We observe that the classification performance of all LLMs has improved significantly when evaluated on this corrected dataset. The average F1 score increased from 0.848 to 0.897, representing a gain of 0.05 points. Likewise, the average accuracy rose by approximately five percentage points.

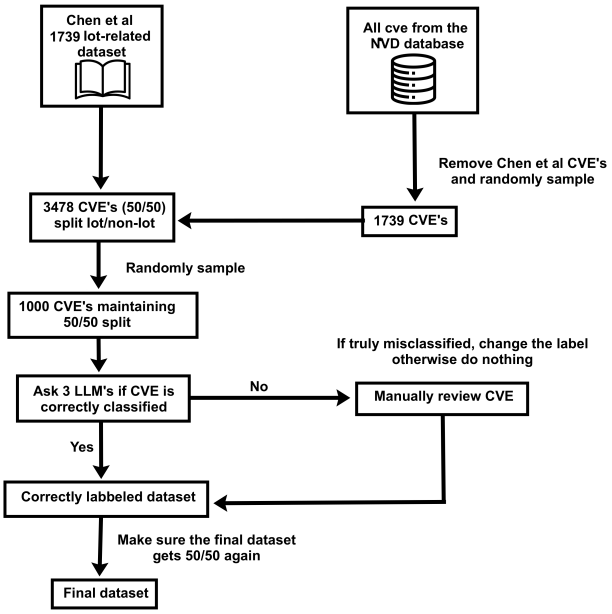

Fig. 2. Dataset construction overview.

This suggests that the original dataset contained mislabeled entries, and that the LLMs had previously made correct predictions which were penalized due to inaccurate ground-truth labels.

4.6 LLM vs. fine-tuned classifiers

To answer RQ2, we fine-tune the selected classifiers on the training set and evaluate their performance on the test set, both before and after applying the manual review corrections.

Tables 2 and 4 present the performance of the fine-tuned classifiers on the original and corrected test sets, respectively. All classifiers perform well and show similar results, with SciBERT slightly outperforming the others, achieving up to 1% higher accuracy. Comparing Table 1 and Table 2, which contrast LLMs with fine-tuned classifiers on the original test set, we observe that the fine-tuned classifiers significantly outperform the LLMs, achieving accuracies of 96–97%, compared to a maximum of 87% for the LLMs. However, this performance gap narrows when evaluating on the corrected test set, as shown in Tables 3 and 4. While classifiers still outperform LLMs, their accuracy drops to 94–95%, whereas LLMs reach up to 92%. This reduction in classifier performance is likely due to the fact that they were trained on the original dataset, which contained labeling noise. As a result, the models were optimized on misclassified data, leading to decreased accuracy when evaluated on the manually corrected test set.

Another notable difference lies in execution time. The training time for the fine-tuned classifiers was relatively short, averaging around 5-9 minutes, with evaluation taking only a few hundred milliseconds. In contrast, LLMs exhibited significantly higher inference times, ranging from 1.3 seconds to over 15 seconds per CVE, depending on model size and model type.

Although in our experiments the fine-tuned classifiers consistently outperform the LLMs in terms of accuracy and efficiency, some considerations should be noted. First, the presence of incorrect labels in the original dataset highlights the need for a

Table 3. Classification performance for different LLMs on the testset after review.

Model	Accuracy	Precision	Recall	F1 Score	Avg time per sample (s)
Gemma (12B)	**92.4%**	0.953	0.898	**0.924**	8.4
LLaMA 3.1 (8B)	91.3%	**0.975**	0.847	0.907	1.6
Mistral (7B)	92.0%	0.923	0.917	0.920	1.5
DeepSeek-R1 (7B)	89.1%	0.894	0.887	0.890	15.9
Gemma 3 (4B)	89.2%	0.870	0.921	0.895	**1.3**
Phi-4 Mini (3.8B)	82.5%	0.761	**0.948**	0.844	2.2

Table 4. Classification performance for fine-tuned Transformers on the testset after review.

Model	Accuracy	Precision	Recall	F1 Score	Avg time per sample (s)
SciBERT (scivocab-uncased)	**95.7%**	**0.990**	0.969	**0.980**	0.008
DeBERTa v3 Small	94.8%	0.943	**0.990**	0.966	**0.005**
DistilBERT (base-uncased)	95.4%	0.985	0.955	0.970	0.007

high-quality dataset of IoT-related CVEs which is almost overlooked in the literature. Second, one of the key strengths of LLMs lies in their ability to operate without any task-specific fine-tuning as demonstrated in prior work [6]. Unlike classifiers that rely on labeled training data to learn explicit patterns, LLMs leverage their broad pre-trained knowledge to reason over the content, enabling them to interpret subtle semantic cues and contextual information that may not align with previously seen examples. Furthermore, in our evaluation setup, the test samples closely resemble the training data, further favoring the performance of fine-tuned models. For example, a CVE describing a vulnerability in a newly released IoT-connected medical device may contain unfamiliar terms or product names. A fine-tuned classifier might fail to classify it as IoT-related due to lack of prior exposure. In contrast, an LLM can infer the relevance of IoT from signals such as remote access or embedded firmware [25], even without recognizing the specific product. These findings suggest that while fine-tuned classifiers excel under stable and well-annotated data, LLMs offer valuable robustness and adaptability in scenarios where label quality or domain familiarity is limited. Note that, although the execution time of LLMs is significantly higher than that of the classifiers, it may be acceptable given that labeling is a one-time process.

4.7 Feature Importance

To address RQ3, we initially prompted the LLMs themselves to identify which input fields they considered most important for their classification decisions. When providing the CVE data to each model, we included an additional instruction in the prompt asking the model to specify which field contributed most significantly to its decision. This approach enables us to directly analyze which CVE fields the models perceive as most informative for determining IoT relevance.

Table 5 shows the resutls. These findings should be interpreted with caution, as not all fields are consistently available across CVEs, particularly in older entries. Only description, published date, and provider short name are present for all samples, while fields like affected items and tags are often missing. To provide context, the table includes field presence counts indicating how frequently each field was available during evaluation. To account for these differences across fields, we normalize the counts by the number of CVEs in which each field was present. The normalized selection rate is computed as:

$$\text{Selection Rate} = \frac{\text{Field Selection Count}}{\text{Field Presence Count}}.$$

The normalized results for all fields are shown in Table 6. A clear trend emerges: When asked themselves, most models identify affected_items as the primary contributor, followed by description, whereas published_date is unsurprisingly never selected given its limited relevance to IoT classification.

Notably, model size appears to influence field preference. Smaller models, such as *Phi-4 Mini (3.8B)* and *Gemma 3 (4B)*, said they rely predominantly on description and affected_items, with only a single selection of provider_short_name. In contrast, larger models consistently prioritize affected_items, suggesting that their increased capacity enables more in-depth exploitation of structured fields beyond simple descriptions.

While these self-reported attributions provide insight into how models perceive field relevance, they are inherently limited: LLMs still receive the full CVE input, including all contextual information, making it difficult to isolate the true impact of individual fields on classification decisions, though it remains interesting to observe how the models interpret their own reasoning.

To better understand which input fields influence model predictions, we also adopt a SHAP-inspired post-hoc explanation approach, following the methodology described in [5]. Traditional SHAP relies on internal model signals such as gradients or probability distributions, which are not accessible in large language models due to their black-box nature. Therefore, we use a black-box approach to measure each field's contribution. We focus on the two most important fields, namely affected_items and description, as identified by the LLMs in their self-reported importance assessments. To evaluate the impact of these fields, we test all three combinations of input:

- Only description
- Only affected_items
- Both fields combined

For each configuration, we record whether the model correctly classifies the CVE. This enables us to assess the contribution of each input field and understand how they influence predictions both individually and in combination. The analysis is conducted on the 812 CVEs where both fields are present, and the results are presented in Figure 3.

We observe that all LLMs achieve the highest classification performance when both description and affected_items are provided.

Table 5. Most Significant Fields contributing to LLM classification decisions.

Model	affected_items	description	provider_short_name	published_date	tags
Gemma (12B)	493	394	29	0	0
LLaMA 3.1 (8B)	668	226	22	0	0
Mistral (7B)	516	389	11	0	0
DeepSeek-R1 (7B)	504	392	19	0	0
Gemma 3 (4B)	228	687	1	0	0
Phi-4 Mini (3.8B)	345	571	0	0	0
Field Presence Count	**812**	**916**	**916**	**916**	**836**

Table 6. Normalized Selection Rate per Field (relative to field presence, values between 0 and 1).

Model	affected_items	description	provider_short_name
Gemma (12B)	0.6071	0.4300	**0.0317**
LLaMA 3.1 (8B)	**0.8228**	0.2467	0.0240
Mistral (7B)	0.6356	0.4247	0.0120
DeepSeek-R1 (7B)	0.6207	0.4279	0.0208
Gemma 3 (4B)	0.2807	**0.7500**	0.0011
Phi-4 Mini (3.8B)	0.4249	0.6235	0.0000

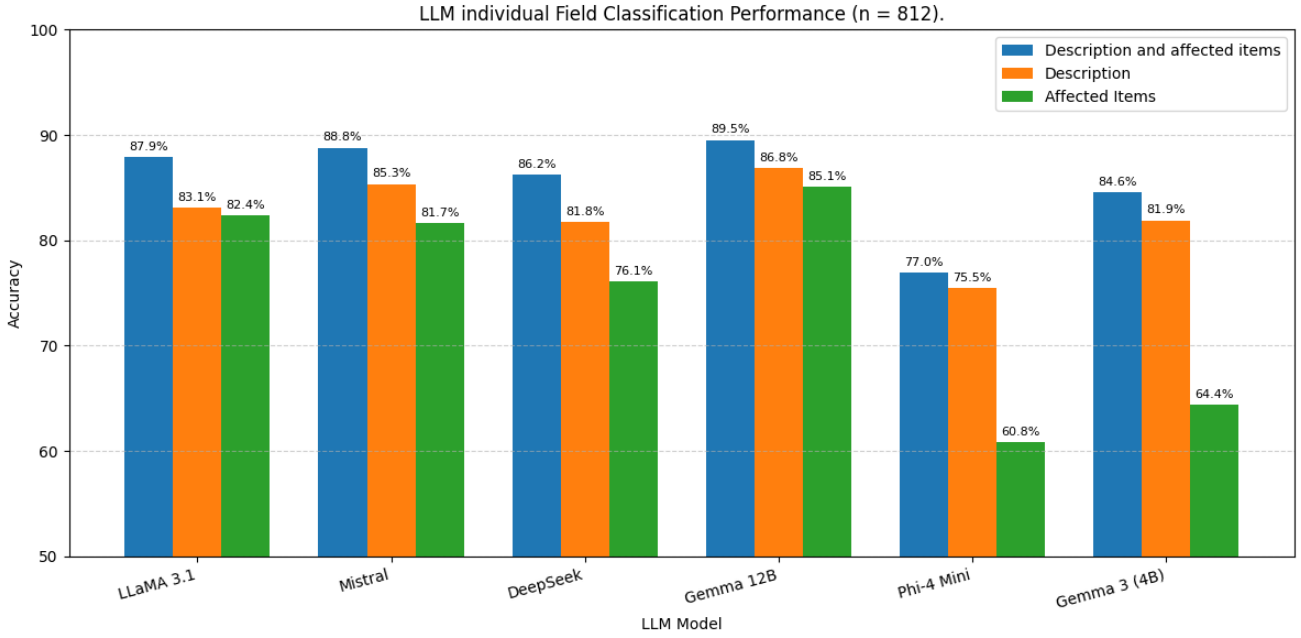

Fig. 3. The accuracy of various LLMs over different set of CVE fields.

Among the individual fields, description consistently outperforms affected_items, especially in smaller models like *Phi-4 Mini (3.8B)* and *Gemma (4B)*. For instance, *Gemma (4B)* drops all the way to 64.4% accuracy when relying solely on affected_items, suggesting difficulty in extracting meaningful patterns without the richer context that description provides.

Larger models show a smaller gap between the two fields, reflecting a greater ability to infer IoT relevance even from more limited input. Notably, although most models identified affected_items as the most important field when asked directly, their actual performance reveals a stronger dependence on description. This mismatch likely arises because description often embeds references to affected products along with additional contextual cues, providing signals that enhance classification. When explicitly asked to reflect, models may focus on concrete identifiers like product names, even if their predictions are more strongly driven by descriptive context.

Another noteworthy observation is that classification performance using only the description and affected_items fields is approximately 2–3 percentage points lower compared to when all input fields from Section 3.3 are provided, as shown in Table 3. This suggests that, while the additional fields may not individually offer strong contextual signals for IoT relevance, they still contribute meaningfully when combined with the core fields. The models leverage all available inputs to achieve higher classification accuracy.

Based on these observations, we conclude that description is the most important field for IoT classification. The isolated results show a consistent pattern where description outperforms affected_items. However, the two fields are complementary when used together.

5 DISCUSSION

Building on the results, several important aspects emerge that further illustrate the strengths, limitations, and practical considerations. We discuss these aspects in detail in the following sections.

5.1 Flexibility and Adaptability

A major strength of large language models is their flexibility. Unlike traditional classifiers that rely on explicit features and task-specific training, LLMs can generalize across diverse inputs with little or no fine-tuning. However, we have seen that this flexibility comes at a computational cost. LLMs take significantly longer to evaluate each CVE, often several seconds more compared to fine-tuned models. Still, given that only around 108 CVEs are published daily [10], this overhead is rarely a practical bottleneck. Since classification would typically be applied after release, the slightly longer evaluation time per CVE remains acceptable, especially given the benefit of deeper contextual understanding that LLMs provide. While LLMs may not be ideal for scenarios requiring extremely high throughput, their computational demands are manageable for most practical applications in IoT-related CVE assessment.

5.2 Handling missing data

Another important distinction and advantage in LLMs lies in how both model types handle variability and incompleteness in the input data. The fine-tuned classifiers rely heavily on the availability and consistency of structured fields during both training and inference. When these fields are missing or incomplete, which is common in CVE databases, their performance degrades, as they have limited capacity to compensate for missing signals. In contrast, LLMs are inherently more resilient to such data sparsity. Even when structured fields are unavailable, LLMs can rely on the unstructured text contained in CVE descriptions and extract contextual clues to guide their predictions. This ability to flexibly adapt to incomplete inputs makes LLMs particularly more well-suited for real-world vulnerability datasets, where data quality and field completeness often vary widely across records.

5.3 Limitations

While this study offers useful insights, several limitations should be noted. The manually labeled dataset consists of 916 CVE entries, which reflects practical constraints in time and resources for manual review. Although this sample allows for meaningful comparisons between models, a larger dataset would help strengthen the conclusions and better capture the full diversity of IoT-related vulnerabilities, including more edge cases and less common device types.

It should also be noted that the LLMs in this study were evaluated in a zero-shot setting, where each CVE was independently processed without any form of iterative learning or accumulated feedback. Although each prompt included our definition of what we consider an IoT related CVE, along with examples, the models did not benefit from adjusting based on prior classifications or feedback. If LLMs could incorporate feedback or learn from previous predictions during operation, their performance would likely improve further, particularly for more ambiguous or edge-case CVEs. Exploring such adaptive or semi-supervised approaches may offer an interesting direction for future work, potentially narrowing the gap between LLMs and fine-tuned classifiers.

Another limitation lies in the manual review process itself. The labeling was performed by a single reviewer. From a scientific and methodological standpoint, it is standard practice to involve at least three independent reviewers who each assess the CVEs and apply a majority voting rule to categorize them. This helps reduce individual bias and improve the overall quality and reliability of the dataset. The absence of such a process in this study means that the corrected labels, while carefully assigned, may still reflect individual interpretation.

Finally, it is important to recognize that the study ultimately depends on the quality and consistency of the underlying NVD database itself. Although the NVD serves as the most widely used public source for vulnerability information, it is not immune to issues such as inconsistent data entry or incomplete records. These factors can introduce noise into both training and evaluation, potentially influencing model behavior in subtle ways. As a result, part of the observed model performance may reflect not only the model capability but also the structure and limitations of the data source itself.

6 CONCLUSION

In this paper, we explored the use of large language models for classifying IoT-related vulnerabilities within the CVE dataset. While fine-tuned classifiers trained on domain-specific data achieved the highest overall performance, the difference compared to the best-performing LLMs was relatively modest. Importantly, LLMs were able to achieve strong results without any task-specific fine-tuning, relying solely on their pre-trained knowledge to interpret the CVE descriptions.

Among the evaluated models, we found that larger LLMs, in particular *Gemma 3 (12B)*, consistently delivered the strongest performance across most metrics. This suggests that model scale plays a key role in improving LLM classification capability for this task. However, our evaluation was limited to models within a certain size range, and it remains likely that even larger models could further improve classification accuracy. In addition, our feature importance analysis showed that the description field was the most critical across all models. However, incorporating additional fields further improves classification accuracy by providing complementary context.

Overall, our findings indicate that LLMs, especially larger ones can serve as effective classifiers for IoT-related CVE data, even in zero-shot settings. This makes them a practical and flexible alternative for automated vulnerability classification, particularly in scenarios where labeled data is limited or evolving rapidly.

ACKNOWLEDGMENTS

I would like to express gratitude to my supervisor, Tina Rezaei, for her guidance, support, and insightful feedback throughout this research. I am also sincerely thankful to Dr. Suzan Bayhan for her lectures and which greatly contributed to my academic growth.

REFERENCES

[1] Harika Abburi, Michael Suesserman, Nirmala Pudota, Balaji Veeramani, Edward Bowen, and Sanmitra Bhattacharya. 2023. Generative AI Text Classification using Ensemble LLM Approaches. , 104–115 pages. https://arxiv.org/abs/2309.07755

[2] DeepSeek AI. 2024. DeepSeek R1: An Open, Efficient, and Strong Language Model. Available at: https://huggingface.co/deepseek-ai/deepseek-llm-7b-instruct.

[3] Mistral AI. 2023. Introducing Mistral 7B. Available at: https://mistral.ai/news/introducing-mistral-7b/.

[4] Timon Beld. 2024. IoT CVE Classification Dataset. https://github.com/MrGainn/cve_lmm_data Available at: https://github.com/MrGainn/cve_lmm_data.

[5] Ahsan Bilal, David Ebert, and Beiyu Lin. 2025. LLMs for Explainable AI: A Comprehensive Survey. *ACM Transactions on Intelligent Systems and Technology* (March 2025). https://arxiv.org/pdf/2504.00125v1.

[6] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems* 33 (2020), 1877–1901. https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[7] Xiang Chen, Changlin Yang, Yuhong Nan, and Zibin Zheng. 2024. An Empirical Study of High-Risk Vulnerabilities in IoT Systems. *IEEE Internet of Things Journal* (2024), 1590–1601. https://ieeexplore.ieee.org/document/10769424

[8] Brittany D. Davis, Janelle C. Mason, and Mohd Anwar. 2020. Vulnerability Studies and Security Postures of IoT Devices: A Smart Home Case Study. *IEEE Internet of Things Journal* 7, 10 (2020), 10200–10208. https://ieeexplore.ieee.org/document/9050664

[9] Google DeepMind. 2024. Gemma: Open Models from Google DeepMind. Available at: https://ai.google.dev/gemma.

[10] Jerry Gambling. 2025. 2024 CVE Data Review. https://jerrygamblin.com/2025/01/05/2024-cve-data-review/. Accessed: 2025-06-16.

[11] GeeksforGeeks. 2022. 3-layer IoT Architecture. https://www.geeksforgeeks.org/3-layer-iot-architecture/. Accessed: 2025-06-24.

[12] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. 2013. Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions. *Future Generation Computer Systems* 29, 7 (2013), 1645–1660. https://doi.org/10.1016/j.future.2013.01.010

[13] Sarah Bin Hulayyil and Shancang Li. 2025. An IoT Featureless Vulnerability Detection and Mitigation Platform. *Electronics* 14, 7 (2025), 1459. https://doi.org/10.3390/electronics14071459

[14] Arina Kostina, Lars Müller, and Victor Tan. 2025. Large Language Models For Text Classification: Case Study and Comprehensive Review. *arXiv preprint arXiv:2501.08457* (2025). https://arxiv.org/abs/2501.08457

[15] Satish Kumar Maurya, Om Prakash Pal, and Ketan Sarvakar. 2024. Layered Architecture of IoT. In *Secure and Intelligent IoT-Enabled Smart Cities*. IGI Global. https://doi.org/10.4018/978-1-3693-2373-1.ch009

[16] Meta AI. 2024. LLaMA 3: Open Foundation and Instruction Models. Available at: https://ai.meta.com/llama/.

[17] Microsoft. 2025. Phi-4 Mini Instruct. https://huggingface.co/microsoft/Phi-4-mini-instruct. Released May 1, 2025. Accessed May 2025.

[18] Nataliia Neshenko, Elias Bou-Harb, Jorge Crichigno, Georges Kaddoum, and Nasir Ghani. 2019. Demystifying IoT Security: An Exhaustive Survey on IoT Vulnerabilities and a First Empirical Look on Internet-Scale IoT Exploitations. , 2702–2733 pages. https://ieeexplore.ieee.org/document/8688434

[19] Ollama Team. 2024. Ollama: Run LLMs Locally. https://ollama.com/. Accessed: 2024-06-28.

[20] Kai Sun, Junyi Li, Dian Yu, Zhou Yu, and Kai-Wei Chang. 2023. Text Classification via Large Language Models with Clue-and-Reasoning Prompting. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. Association for Computational Linguistics, 926–942. https://arxiv.org/abs/2305.08377

[21] S. Narasimha Swamy and Solomon Raju Kota. 2020. FDEOS-LP: A Lightweight Protocol Stack for Secure Communication in IoT. *IEEE Access* 8 (2020), 187240–187257. https://ieeexplore.ieee.org/document/9218916

[22] Catherine Tony, Markus Mutas, Nicolás E. Díaz Ferreyra, and Riccardo Scandariato. 2023. LLMSecEval: A Dataset of Natural Language Prompts for Security Evaluations. *arXiv preprint arXiv:2303.09384* (2023). https://arxiv.org/abs/2303.09384 150 natural-language prompts derived from MITRE's Top 25 CWEs.

[23] Bhagyashri Tushir, Hetesh Sehgal, Rohan Nair, Behnam Dezfouli, and Yuhong Liu. 2021. The Impact of DoS Attacks on Resource-Constrained IoT Devices: A Study on the Mirai Attack. arXiv preprint arXiv:2104.09041. https://arxiv.org/abs/2104.09041

[24] Sowmya Vajjala and Shwetali Shimangaud. 2025. Text Classification in the LLM Era – Where Do We Stand? *arXiv preprint arXiv:2502.11830* (Feb 2025). https://arxiv.org/abs/2502.11830

[25] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, and Fei Xia. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv preprint arXiv:2201.11903* (2022). https://arxiv.org/abs/2201.11903

[26] Wikipedia contributors. 2024. Mirai (malware). https://en.wikipedia.org/wiki/Mirai_(malware). Accessed: April 27, 2025.

[27] Binbin Zhao, Shouling Ji, Jiacheng Xu, Yuan Tian, Qiuyang Wei, Qinying Wang, Chenyang Lyu, Xuhong Zhang, Changting Lin, Jingzheng Wu, and Raheem Beyah. 2022. A Large-Scale Empirical Analysis of the Vulnerabilities Introduced by Third-Party Components in IoT Firmware. , 1871–1888 pages. https://dl.acm.org/doi/10.1145/3533767.3534366

A HARDWARE CONFIGURATION

All experiments were conducted on a local workstation with the following hardware specifications:

Table 7. Hardware Configuration

Component	**Specification**
CPU	AMD Ryzen 5 2600X
GPU	NVIDIA GeForce RTX 2060
RAM	64 GB DDR4
Motherboard	MSI B450 Gaming Pro
Storage	Samsung 870 EVO SSD
Power Supply	Corsair RM650e
Operating System	Windows 10 Education