AutoCVSS: Assessing the Performance of LLMs for Automated Software Vulnerability Scoring

Davide Sanvito[1], Giovanni Arriciati[2], Giuseppe Siracusano[1],
Roberto Bifulco[1], Michele Carminati[2]

[1] NEC Laboratories Europe, Heidelberg, Germany
[2] Politecnico di Milano, Milano, Italy

Abstract

The growing volume of daily disclosed software vulnerabilities imposes significant pressure on security analysts, extending the time needed for analysis - an essential step for accurate risk prioritization. Meanwhile, the time between disclosure and exploitation is reducing, becoming shorter than the analysis time and increasing the window of opportunity for attackers. This study explores leveraging Large Language Models (LLMs) for automating vulnerability risk score prediction using the industrial CVSS standard. From our analysis across different data availability scenarios, LLMs can effectively complement supervised baselines in data-scarce settings. In the absence of any annotated data, such as during the transition to new versions of the standard, LLMs are the only viable approach, highlighting their value in improving vulnerability management. We make the source code of AutoCVSS public at `https://github.com/nec-research/AutoCVSS`.

1 Introduction

Over the past 25 years, the Common Vulnerabilities and Exposures (CVE) program established as the de–facto standard to identify and catalog publicly-disclosed software vulnerabilities (CVE, 2024b). In this context, the number of CVE records have steadily increased over the past decade, with a 38% rise from 2023 to 2024 (CVE, 2024a). To help organizations manage this growing volume and effectively assess risk, the National Vulnerability Database (NVD, 2024e) enhances CVE records with severity scores using the Common Vulnerability Scoring System (CVSS), a widely adopted industrial standard (FIRST, 2024b). Security experts, mainly associated with NVD, manually assess the severity of new vulnerabilities by relying on detailed information from CVE records and publicly available data: they compute eight *CVSS Metrics* (FIRST, 2023b), concisely represented as *CVSS Vectors* (an example is provided

in Table 1), and calculate the *Severity Score* on a scale of 0 to 10 (NVD, 2024a). Both the individual metrics and the aggregated severity score are critical components of various stages of vulnerability management, e.g., risk assessment, mitigation planning, and incident response.

However, previous studies (Costa et al., 2022; Aghaei et al., 2022) highlighted a delay of several days between CVE announcements and the publication of corresponding CVSS scores that prevents timely vulnerability management. This issue is exacerbated by the growing volume of published vulnerabilities that significantly increases analysts' burden: the median manual analysis delay increased from 2.7 days in 2019 to 8.2 days in 2023 (Pan et al., 2024). In contrast, attackers are acting faster: the average time to exploit vulnerabilities has decreased sharply, from 44 days in 2019 to just 5 days in 2023 (Google, 2024), with 25% of high-risk vulnerabilities exploited on the day they are published (Qualys, 2023), extending the window of opportunity for attackers.

To mitigate these delays, several works have proposed automating CVSS score assignment for new CVEs using Natural Language Processing (NLP). Most approaches rely on supervised learning, which require annotated data. Although the NVD hosts hundreds of thousands of CVEs (NVD, 2024f), multiple CVSS versions have been released over time and not all records include CVSS values for every version (NVD, 2024a). Specification updates hinder the rapid adaptation of these techniques until sufficient data labeled in the new format becomes available. This issue is particularly critical during version transitions, such as the recent release of CVSS v4.0 (FIRST, 2023c), when annotated data for the new version is scarce.

Large Language Models (LLMs) have demonstrated remarkable performance across various NLP tasks (Yang et al., 2024). Leveraging the textual CVE descriptions, this study explores the

564

feasibility of employing LLM-based approaches to develop a robust and scalable solution for automating CVSS prediction. We evaluate different prompting strategies, spanning zero-shot to few-shot, and leveraging Retrieval-Augmented Generation (RAG) techniques. We consider both open-source and closed-source LLMs, also including the recent Large Reasoning Models (LRMs) variants, in the black-box setting and compare them against fine-tuned BERT supervised baselines[1].

Our study highlights three key findings:
① With abundant annotated data, LLM-based solutions have competitive performance, but they are surpassed by supervised baselines fine-tuned for the specific CVSS prediction task.
② With limited labeled data, supervised baselines based on the most recent encoder-only models perform best. However, when admitting a hybrid approach, LLMs can complement them by offering better prediction performance on half of the CVSS metrics, overall achieving better results.
③ With extremely scarce or unavailable data (e.g., transitioning to v4.0), LLMs remain effective with minimal prompt adaptations to align with the new specification. Although their performance is lower than previous scenarios due to reliance on zero-shot settings, LLMs still aid analysts by outperforming conservative worst-case approaches. Additionally, they facilitate the adoption of new CVSS versions by providing initial predictions, even in the absence of labeled v4.0 data. Then, as more labeled data becomes available, the approach can evolve to more effective hybrid or fully-supervised methods, enhancing accuracy and scalability.

These findings demonstrate the practical potential of LLMs to address vulnerability risk prioritization and their suitability for real-world applications, significantly contributing to automating and streamlining vulnerability management processes. To facilitate reproducibility and further research, we make the source code of AutoCVSS public.

2 CVSS metrics prediction

Several studies considered the prediction of CVSS metrics from CVE descriptions. This paper focuses on predicting the CVSS v3.1 Base Metric group, which evaluates the inherent, time- and environment-independent characteristics of a vulnerability (see Table 1). We classify existing **pre-**

---

[1]In this paper, *LLMs* refer to decoder-only models like GPT, not including encoder-only models like BERT.

Table 1: List of metrics and possible values, sorted by decreasing severity, for the CVSS v3.1 Base Metric group. The CVSS Vector uses the abbreviated metric name reported in parenthesis. For example the CVSS Vector assigned by NVD to CVE-2023-35359 is `CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H`.

CVSS Metric	Labels
Attack Vector (AV)	Network (N), Adjacent (A), Local (L), Physical (P)
Attack Complexity (AC)	Low (L), High (H)
Privileges Required (PR)	None (N), Low (L), High (H)
User Interaction (UI)	None (N), Required (R)
Scope (S)	Changed (C), Unchanged (U)
Confidentiality Impact (C)	High (H), Low (L), None (N)
Integrity Impact (I)	High (H), Low (L), None (N)
Availability Impact (A)	High (H), Low (L), None (N)

diction approaches into three categories.

Individual CVSS Metrics (Cat. I). Like our work, it uses distinct multi-class classification models to predict the eight CVSS metrics. Most approaches leverage transfer learning with pre-trained BERT models (Devlin et al., 2019) or their variants fine-tuned for this task. Shahid and Debar (2021) uses BERT-small, while Costa et al. (2022) and Kühn et al. (2023) employ DistilBERT. Aghaei et al. (2023) applies SecureBERT. Babalau et al. (2021) and Shan et al. (2023) propose multi-task models combining BERT with BiLSTM. Other works combine bag-of-words representations (Elbaz et al., 2020) or word embeddings (Kekül et al., 2024) with traditional Machine Learning (ML) models.

Qualitative CVSS Severity Ratings (Cat. II). It predicts Qualitative Severity Ratings (None, Low, Medium, High, Critical) derived from the 0–10 CVSS score (FIRST, 2023a) as a single 5-class classification task. Kai et al. (2023) and Babalau et al. (2021) fine-tune DistilBERT and BERT-small, respectively. Li et al. (2023) explores prompt learning with BERT and RoBERTa, while Ni et al. (2022) combines fine-tuned BERT with a CNN.

CVSS Severity Score (Cat. III). It directly predicts the CVSS severity score (0–10 range) as a regression task. Pal et al. (2023) employs T5 model (Raffel et al., 2020) in a multi-task setting, including CVSS prediction. Vasireddy et al. (2023), Babalau et al. (2021), and Zhang et al. (2022) instead exploits ML-based models.

2.1 LLM-based CVSS Prediction

Few studies have explored LLMs for predicting CVSS v3.1 metrics, each with notable limitations. McClanahan et al. (2024) evaluates GPT mod-

els (OpenAI, 2024b) to retrieve CVSS scores and vectors for a given CVE ID using only pre-trained knowledge without using its description in input. By design it cannot predict data for new CVEs disclosed after the model's knowledge cutoff date.

CTIBench (Alam et al., 2024) benchmarks ChatGPT and LLaMA3 using zero-shot prompting to predict CVSS vectors. It only tests 1000 samples and ignores supervised approaches.

CVEDrill (Aghaei et al., 2023) compares their fine-tuned SecureBERT models with ChatGPT. The evaluation, including only 100 CVEs, seems to only consider zero-shot prompting.

CVECenter (Luo et al., 2024) predicts CVSS metrics with zero-shot prompting using GPT models leveraging multi-source vulnerability records. Their target is the specialization of the CVSS for different Linux distributions, having the NVD's CVSS in the inputs rather than as predicted output.

Liu et al. (2024) proposes CyberBench, a benchmark for cybersecurity NLP tasks, including CVSS prediction. Although they compared fine-tuned BERT models against few-shots LLMs and fine-tuned LLaMA2 models, their focus is on coarse-grained CVSS severity ratings (Category II).

Isogai et al. (2024) shows that BERT outperforms gpt-4o-mini, with and without fine-tuning, when predicting CVSS v3.1 vectors with zero-shot prompting, with an approach similar to CTIBench.

Positioning of Our Work. Unlike previous studies, our work explores multiple LLM approaches beyond zero-shot prompting, extensively covering closed- and open-source LLMs (also considering recent LRM variants specialized for reasoning), compares supervised baselines (also including latest advances in encoder-only models), and extends evaluations to both CVSS v3.1 and the latest CVSS v4.0 specification, providing a more comprehensive approach and practical assessment of LLMs to the evolving vulnerability scoring needs. Our work belongs to Category I: predicting the individual CVSS metrics has the best value because it enhances transparency by showing which factors contribute to the overall score for better decision-making. In addition, from Category I metrics, we can always derive Categories II and III metrics, whereas the reverse is not possible.

3 AutoCVSS Methodology

Following prior works from Category I (Sec.2), we predict each CVSS metric as an independent multi-
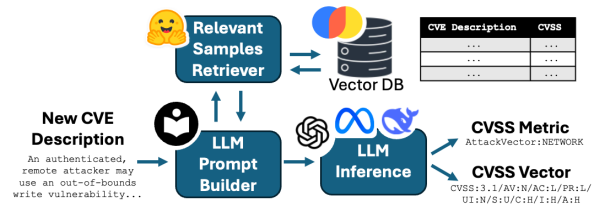

Figure 1: AutoCVSS high-level architecture.

class classification task. We evaluate multiple approaches, considering the closed-source OpenAI GPT-4o (OpenAI, 2024a) and o3-mini (OpenAI, 2025), and open-source Meta LLaMA3 (Grattafiori et al., 2024) and DeepSeek-R1 (DeepSeek-AI, 2025) models. We consider the LLMs general-domain black-box models: LLM fine-tuning for the security domain or CVSS-specific tasks is out of the scope of this paper.[2] We implement the approach depicted in Fig.1 with Instructor (Liu, 2024), an open-source framework for structured output generation with LLMs. We explored both zero-shot and few-shots prompting. Appx. A includes more details for reproducibility and examples of prompts.

Zero-shot Prompting Approaches. They leverage the LLM's inherent knowledge and generalization capabilities without task-specific examples.
Simple Task Description (STD): It relies only on the LLM's pre-trained understanding of CVSS metrics. The task description provides only the name of the CVSS metric to predict and the set of possible labels without any additional context or details.
Detailed Task Description (DTD): It leverages the CVSS specification to enhance the prompt with more context. The task description includes a detailed explanation of the CVSS metric and its labels, directly borrowed from the CVSS Specification Document provided by FIRST (2023b).
Full Vector Prediction (FVP): It directly predicts the entire CVSS vector with a single prompt. Individual CVSS metrics are then extracted with a regular expression. We include this variant as an approach similar to CTIBench (Alam et al., 2024). FVP employ *Chain-of-Thought (CoT)* reasoning (Wei et al., 2022), enabling the LLM to reason step-by-step. This enhances decision-making by dividing complex tasks into smaller, sequential steps, aligning with FVP's holistic approach.

Few-shot Prompting Approaches. They leverage LLMs' in-context learning (ICL) capabili-

---

[2]For completeness, Appx. C includes a preliminary evaluation of LLM fine-tuning, discussing its scope and limitations.

ties (Brown et al., 2020) by including a small set of *(CVE description, label)* examples in the prompt to guide classification. For each test sample, RAG constructs a prompt with the most semantically similar training examples, retrieved through semantic search in a vector database. We use Chroma (2024) to store and query text embeddings generated with a SBERT model (Reimers and Gurevych, 2019). These examples help the LLM predict labels based on similar descriptions. Although all the zero-shot methods can theoretically be extended to few-shot settings, adapting CoT prompting would require extensive manual annotation of reasoning chains, rendering it impractical. Hence, we focus on few-shot variants of *STD* and *DTD*. Also, we exclude LRMs from this approach because their providers suggest that few-shot prompting may produce poor results (Guo et al., 2025; OpenAI, 2024c). In terms of number of training samples (*shots*) for the prompt, we tested values from 1 to 32 and found 24 to have the best cost-improvement trade-off for our task.

To handle cases where the LLM cannot make an informed decision, we introduced an additional label, *Don't Know (DK)*, across all strategies described in this section. This allows the LLM to indicate when the provided description cannot assess the metric. As a final step, samples classified with the *DK* label are mapped to the most severe label for the CVSS metric (e.g., *NETWORK* for *AttackVector*), following the conservative worst-case approach recommended by NVD (2024d) for missing or unclear information.[3] To further examine the implications of consistently applying this approach to *all* the samples, Sec. 4 includes an additional baseline based on this conservative policy.

4 Evaluation

We evaluate the different LLM-based approaches against four baselines, considering three different data availability settings and two CVSS versions.

Baseline approaches. We selected two related works from the literature, both based on fine-tuned BERT models: DistilBERT-E (Costa et al., 2022) and CVEDrill (Aghaei et al., 2023). DistilBERT-E relies on DistilBERT (Sanh et al., 2019), a smaller and faster version of BERT, to predict the CVSS metric using eight separate multi-class classification models. CVEDrill adopts a similar approach, with eight models, but employs Se-

cureBERT (Aghaei et al., 2022), a cybersecurity-specific variant of BERT built on RoBERTa (Liu et al., 2019) and pre-trained on a large corpus of cybersecurity data. As an additional baseline, we also fine-tuned ModernBERT models, a recent general-domain state-of-the-art encoder-only model (Warner et al., 2024). Finally, we added the Worst Case Label (WCL) baseline, which assigns each sample the most severe label for each CVSS metric, following the conservative strategy described in Sec. 3.

Evaluation Metrics. We adopt the same evaluation metrics used in related works (Category I, Sec. 2), namely Accuracy (A), weighted F1-score (wF1), and Macro F1-score (MF1). Although our work does not directly predict the CVSS severity score, we extend our evaluation to also align with works from Categories II and III. From the eight predicted metrics, we compute the severity score (NVD, 2024b) and report the Mean Average Error (MAE) and Mean Squared Error (MSE). Additionally, we quantize the computed severity score into the Qualitative Severity Rating Scale (QSRS) and discuss the results in detail in Appendix E.

Dataset. For our evaluation, we used all the public CVE records published in the NVD between January 2023 and April 2024, retaining only those with CVSS v3.1 information released by the NVD and obtaining a dataset of over 27k CVE records. Fig. 2 illustrates the class proportions for the eight CVSS Metrics and the distributions for the severity scores values and for the CVE description lengths, in characters. We perform a stratified 80/20 train/test split on the dataset. We stratify the split by the CVSS vector to ensure that (1) class proportions are consistent in both sets and (2) each CVE ID appears exclusively in either the train or test set for all eight models. Performances are computed on the test set, while the train set is used either to fine-tune models (for supervised baselines) or to populate the Vector Store (for few-shot LLM-based methods).

4.1 Full dataset evaluation

Table 2 summarizes the performance metrics, averaged across the eight CVSS metrics, when considering the entire dataset. For the 0-shot approaches, we only report the top-6 configurations:[4] closed-source models offer the best performance with LRM variants better than LLM. Although the best 0-shot configurations, dominated by o3-mini, show

---

[3]Although the DK-labeled samples are negligible, this mapping is beneficial in most of the cases (cf. Appx. B).

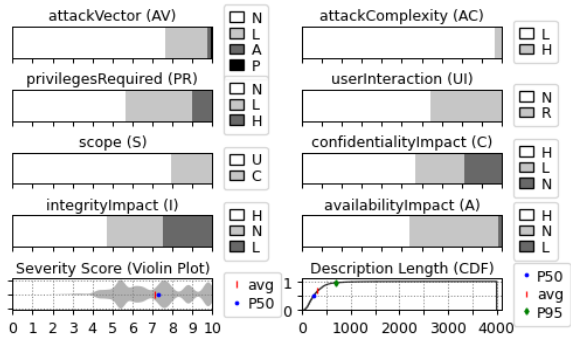[4]Table 6 in Appx. D includes the remaining configurations.

Figure 2: Dataset statistics: class proportions for the eight CVSS Metrics and distributions for the Severity Scores values and for the CVE Description Lengths.

Table 2: Full dataset evaluation. Top-3 values are marked in **bold**, ***italic bold*** and *italic*, respectively. LLMs use *[Prompt Strategy][Model]* naming (Sec. 3).

Method	A	wF1	MF1	MAE	MSE
Worst Case Label	0.595	0.466	0.276	2.836	11.094
DistilBERT-E	0.903	0.897	0.739	0.762	1.983
ModernBERT	*0.928*	*0.927*	*0.858*	*0.554*	*1.270*
CVEDrill	**0.929**	**0.928**	**0.858**	**0.549**	**1.231**
0s FVP o3-mini	0.846	0.836	0.722	1.200	3.991
0s FVP GPT-4o	0.754	0.743	0.643	1.655	4.861
0s STD o3-mini	0.804	0.793	0.676	1.609	6.000
0s STD GPT-4o	0.757	0.765	0.651	1.317	3.343
0s DTD o3-mini	0.767	0.755	0.654	2.095	8.747
0s DTD GPT-4o	0.750	0.751	0.634	1.494	4.573
24s STD GPT-4o	*0.915*	*0.915*	*0.842*	*0.608*	*1.469*
24s STD LLaMA3	0.905	0.903	0.825	0.667	1.623

that FVP works best, when moving to few-shots approaches, we had to limit to STD and DTD, only considering LLMs without LRMs (cf. Sec. 3). We thus picked the best among STD and DTD, i.e., STD, for both GPT-4o and LLaMA3 models.

Results indicate that the best LLM-based approach (GPT-4o few-shots) surpasses DistilBERT-E, but falls short of ModernBERT and CVEDrill. Table 10 in Appx. F breaks down numbers by CVSS metric. The stronger performance of ModernBERT and CVEDrill can be attributed to their larger model size (150 and 125 million parameters, nearly double DistilBERT-E's 65 million). In addition, CVEDrill also received a pre-training on a cybersecurity-specific corpus before fine-tuning. The Worst Case Label baseline, serving as the lower bound, delivers the poorest performance, significantly underperforming even the worst LLM-based baseline. As noted in Sec.4, we also report the MAE and MSE of the severity score calculated from the eight predicted metrics, observing that the general ranking of methods is preserved. A similar outcome is observed when analyzing the results in

Table 3: Low-resource setting. Top-3 values are marked in **bold**, ***italic bold*** and *italic*, respectively. Notice that the marking of top values excludes the hybrid approach.

Method	A	wF1	MF1	MAE	MSE
Worst Case Label	0.600	0.470	0.278	2.798	11.145
DistilBERT-E	0.869	0.858	0.673	0.899	2.334
ModernBERT	**0.921**	**0.920**	**0.848**	**0.631**	**1.767**
CVEDrill	0.887	0.876	0.695	0.868	2.308
24s STD GPT-4o	*0.909*	*0.908*	*0.833*	*0.655*	*1.835*
24s STD LLaMA3	*0.896*	*0.893*	*0.821*	*0.715*	*2.013*
SL+LLM Hybrid	0.922	0.920	0.860	0.677	1.894

terms of QSRS (Cat. III metrics, cf. Appx. E).

Our results align with recent findings (Yang et al., 2024) and validate them for our CVSS prediction task: while both LLMs and fine-tuned models perform well with abundant annotated data, fine-tuned models generally outperform LLMs in traditional NLU tasks like text classification. The authors also suggest that LLMs may outperform fine-tuned models in scenarios with limited annotated data. In the next section, we explore this aspect for our task by considering a low-resource setting, with limited train data for fine-tuning and for the Vector Store.

4.2 Low-resource setting evaluation

In this setting, we considered a subset of the dataset by only including the 2.1k vulnerabilities disclosed in 2024[5] and evaluated the same set of methods from Sec. 4.1. Since the previous considerations for zero-shot approaches from Section 4.1 still apply, for the sake of brevity, we omit their values here and report them in Table 7 in Appendix D.

Table 3 shows that the best-performing LLM configurations are now just behind ModernBERT, which proves to be more robust to the reduced fine-tuning data compared to other supervised baselines. A similar ranking, with both few-shots LLMs entering the top-3 positions, can be observed when analyzing the results in terms of QSRS (cf. Appx. E). Although ModernBERT has the best-averaged performance, it is outperformed by LLMs on specific CVSS metrics (cf. Table 11 in Appx. F). By envisioning a scenario where each metric is predicted by a different method, LLMs can complement ModernBERT by offering better performance on 4 out of 8 metrics, overall achieving the best results ("SL+LLM Hybrid" configuration in Table 3).

This low-resource setting is particularly relevant when considering the limited availability of labeled

---

[5]The dataset, roughly 10% of the original size, is again divided into train/test sets with a stratified 80/20 splitting.

Table 4: CVSS v4.0. Top-3 values are marked in **bold**, ***italic bold*** and *italic*, respectively.

Method	A	wF1	MF1	MAE	MSE
Worst Case Label	0.560	0.437	0.250	2.305	7.528
0s FVP o3-mini	0.701	0.684	0.519	1.268	***2.763***
0s STD o3-mini	*0.715*	*0.710*	**0.564**	**1.187**	2.596
0s DTD o3-mini	**0.780**	**0.765**	**0.586**	2.103	9.970
0s DTD DeepSeek	***0.739***	***0.727***	0.536	**1.179**	3.309
0s DTD GPT-4o	0.691	0.710	*0.547*	*1.242*	*2.975*
0s DTD LLaMA3	0.689	0.697	0.522	1.984	7.969

data, like in the transition from CVSS v3.1 to v4.0. In this phase, hybrid solutions combining supervised methods and LLMs could perform better than supervised ones alone. CVSS v4.0 was officially launched in General Availability (GA) in November 2023 (FIRST, 2023c), but at the time of writing, more than one year later, just a few CVE records include v4.0 data. In the next section, we investigate whether LLMs in a zero-shot setting can provide a good enough solution for the challenging setting of a complete lack of labeled data, which prevents a priori the adoption of both the supervised and the LLM few-shot methods.

As additional evaluation, in Appx. G we considered a subset of test set CVEs whose CVSS has been released after LLMs' knowledge cutoff dates. Despite this temporal holdout, results are in line with those observed in Section 4.1 and in this section, suggesting that for this task the performance of LLMs should not be attributed to the memorization of specific examples the LLM may have encountered during its pre-training process.

4.3 Towards CVSS v4.0

CVSS v4.0 introduces key updates: a new *attack-Requirements* metric, removal of the *scope* metric, updates to *userInteraction* labels, and additional *Impact* metrics, totaling 11 metrics. To the best of our knowledge, this is the first practical evaluation of CVSS v4.0. The lack of sufficient labeled samples calls for a zero-shot setting, but the CVSS v4.0 Specification Document (FIRST, 2023d) facilitates the prompts' adaptation. The NVD provides minimal data for CVSS v4.0, but FIRST (2024a) offers a supplementary document with 38 annotated CVE records suitable as test data in our evaluation.

Table 4 includes the top-6 configurations[6] and shows that, in contrast to previous results, here LRMs with a DTD approach are the two best-performing configurations. This can be attributed

to the GA date closely aligning with the LLMs' knowledge cutoff dates[7], as well as the limited adoption of the new standard, which likely results in poor CVSS v4.0 knowledge in the pre-training data. In this case, a more detailed task description in the prompt, combined with the enhanced reasoning capabilities of LRMs, offer the best performance. The different ranking for MAE/MSE may stem from the CVSS scoring formula, which assigns different weights to metrics (NVD, 2024c).

Although performance in absolute terms is lower than in previous sections, especially due to the updated metrics (cf. Table 12 in Appendix F), the results demonstrate that LLMs can produce an initial set of labels with performance up to 33 percentage points higher than the most conservative baseline (WCL). This provides a valuable tool to support the adoption of the new release. As more labeled data becomes available, the process can transition to a low-resource scenario, enabling more effective hybrid approaches, and, eventually, more powerful fully-supervised methods. Considering that the minor version update from v3.0 to v3.1 took several months to reach the first 1,000 samples post-announcement, the adoption timeline for a major version update like v4.0 is likely to be even longer.

5 Conclusions

The current cybersecurity landscape, with a growing number of vulnerabilities disclosed and attackers acting faster, poses a significant challenge for analysts to keep up with the emerging threats. This work explored the potential of Large Language Models (LLMs) for automating CVSS prediction across different data availability settings. Our findings demonstrate that when abundant labeled data is available, LLMs are competitive, but are surpassed by supervised approaches. LLMs are, instead, a valuable solution in scenarios of scarce data availability, complementing the best supervised baseline for half of the CVSS metrics. Finally, in extreme data scarcity, such as during transitions to new CVSS versions, LLMs are the only viable approach and can provide better predictions than conservative approaches and support the adoption of updated scoring systems. These findings highlight LLMs' practical potential to enhance vulnerability risk prioritization. Future works will expand predictions to consider temporal and environmental factors for a more comprehensive risk assessment.

---

[6]Table 8 in Appx. D includes the remaining configurations.

[7]Oct. and Dec. 2023 for OpenAI and Meta models used.

Limitations

This study has two main limitations. First, our evaluation is based on specific LLM snapshots, detailed in Appx. A. Given the pace of advancement in LLMs, newer and more powerful LLMs may provide improved performance for AutoCVSS without requiring significant changes to the proposed system. Second, due to the scarce availability of labeled data, the CVSS v4.0 evaluation is restricted to LLMs in the zero-shot setting. As the adoption of the latest release increases, the availability of a larger amount of labeled data would enable a more comprehensive evaluation including the comparison against supervised baselines and few-shots LLMs approaches.

References

Ehsan Aghaei, Ehab Al-Shaer, Waseem Shadid, and Xi Niu. 2023. Automated cve analysis for threat prioritization and impact prediction. *arXiv preprint arXiv:2309.03040*.

Ehsan Aghaei, Xi Niu, Waseem Shadid, and Ehab Al-Shaer. 2022. Securebert: A domain-specific language model for cybersecurity. In *International Conference on Security and Privacy in Communication Systems*, pages 39–56. Springer.

Md Tanvirul Alam, Dipkamal Bhushl, Le Nguyen, and Nidhi Rastogi. 2024. Ctibench: A benchmark for evaluating llms in cyber threat intelligence. *arXiv preprint arXiv:2406.07599*.

Ion Babalau, Dragos Corlatescu, Octavian Grigorescu, Cristian Sandescu, and Mihai Dascalu. 2021. Severity prediction of software vulnerabilities based on their text description. In *2021 23rd international symposium on symbolic and numeric algorithms for scientific computing (SYNASC)*, pages 171–177. IEEE.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Joana Cabral. 2022. GitHub - CVSS Prediction.

Chroma. 2024. Chroma: The AI-Native Open-Source Embedding Database.

Joana Cabral Costa, Tiago Roxo, João BF Sequeiros, Hugo Proenca, and Pedro RM Inacio. 2022. Predicting cvss metric via description interpretation. *IEEE Access*, 10:59125–59134.

CVE. 2024a. CVE Metrics.

CVE. 2024b. CVE Program.

DeepSeek-AI. 2025. DeepSeek-R1-Distill-Llama-70B (Ollama).

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics.

Clément Elbaz, Louis Rilling, and Christine Morin. 2020. Fighting n-day vulnerabilities with automated cvss vector prediction at disclosure. In *Proceedings of the 15th International Conference on Availability, Reliability and Security*, pages 1–10.

FIRST. 2023a. CVSS Qualitative Severity Rating Scale.

FIRST. 2023b. CVSS v3.1 Specification Document.

FIRST. 2023c. CVSS v4.0 Press Release.

FIRST. 2023d. CVSS v4.0 Specification Document.

FIRST. 2024a. CVSS v4.0 Examples.

FIRST. 2024b. Forum of Incident Response and Security Teams.

Google. 2024. How Low Can You Go? An Analysis of 2023 Time-to-Exploit Trends.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hugging Face. 2021. sentence-transformers/all-MiniLM-L6-v2.

Hugging Face. 2023. ehsanaghaei/SecureBERT.

Hugging Face. 2024a. answerdotai/ModernBERT-base.

Hugging Face. 2024b. meta-llama/Meta-Llama-3.1-70B-Instruct.

Sho Isogai, Shinpei Ogata, Yutaro Kashiwa, Satoshi Yazawa, Kozo Okano, Takao Okubo, and Hironori Washizaki. 2024. Toward extracting learning pattern: A comparative study of gpt-4o-mini and bert models in predicting cvss base vectors. In *2024 IEEE 35th International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 127–134. IEEE.

Shaofeng Kai, Fan Shi, and Jinghua Zheng. 2023. Vuldistilbert: A cps vulnerability severity prediction method based on distillation model. *Security and Communication Networks*, 2023(1):2118305.

Hakan Kekül, Burhan Ergen, and Halil Arslan. 2024. Estimating vulnerability metrics with word embedding and multiclass classification methods. *International Journal of Information Security*, 23(1):247–270.

Philipp Kühn, David N Relke, and Christian Reuter. 2023. Common vulnerability scoring system prediction based on open source intelligence information sources. *Computers & Security*, 131:103286.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Xiangwei Li, Xiaoning Ren, Yinxing Xue, Zhenchang Xing, and Jiamou Sun. 2023. Prediction of vulnerability characteristics based on vulnerability description and prompt learning. In *2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 604–615. IEEE.

Jason Liu. 2024. Instructor: A library for structured outputs from large language models.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zefang Liu, Jialei Shi, and John F Buford. 2024. Cyberbench: A multi-task benchmark for evaluating large language models in cybersecurity.

Jing Luo, Heyuan Shi, Yongchao Zhang, Runzhe Wang, Yuheng Shen, Yuao Chen, Xiaohai Shi, Rongkai Liu, Chao Hu, and Yu Jiang. 2024. Cvecenter: Industry practice of automated vulnerability management for linux distribution community. In *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering*, pages 329–339.

Kylie McClanahan, Sky Elder, Marie Louise Uwibambe, Yaling Liu, Rithyka Heng, and Qinghua Li. 2024. When chatgpt meets vulnerability management: the good, the bad, and the ugly. In *2024 International Conference on Computing, Networking and Communications (ICNC)*. IEEE.

Xuming Ni, Jianxin Zheng, Yu Guo, Xu Jin, and Ling Li. 2022. Predicting severity of software vulnerability based on bert-cnn. In *2022 International Conference on Computer Engineering and Artificial Intelligence (ICCEAI)*, pages 711–715. IEEE.

NVD. 2024a. NVD - CVE FAQs.

NVD. 2024b. NVD - CVSS v3 Calculator.

NVD. 2024c. NVD - CVSS v4 Calculator.

NVD. 2024d. NVD - General FAQs.

NVD. 2024e. NVD - Home.

NVD. 2024f. NVD Dashboard.

Ollama. 2025. Ollama.

OpenAI. 2024a. OpenAI GPT-4o System Card.

OpenAI. 2024b. OpenAI Models documentation.

OpenAI. 2024c. OpenAI o1 series - Reasoning models.

OpenAI. 2025. OpenAI o3-mini System Card.

Kuntal Kumar Pal, Kazuaki Kashihara, Ujjwala Anantheswaran, Kirby C Kuznia, Siddhesh Jagtap, and Chitta Baral. 2023. Exploring the limits of transfer learning with unified model in the cybersecurity domain. *arXiv preprint arXiv:2302.10346*.

Shengyi Pan, Lingfeng Bao, Jiayuan Zhou, Xing Hu, Xin Xia, and Shanping Li. 2024. Towards more practical automation of vulnerability assessment. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pages 1–13.

Qualys. 2023. 2023 Threat Landscape Year in Review: If Everything Is Critical, Nothing Is.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Mustafizur R Shahid and Hervé Debar. 2021. Cvss-bert: Explainable natural language processing to determine the severity of a computer security vulnerability from its description. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1600–1607. IEEE.

Chun Shan, Ziyi Zhang, and Siyi Zhou. 2023. A multi-task deep learning based vulnerability severity prediction method. In *2023 IEEE 12th International Conference on Cloud Networking (CloudNet)*, pages 307–315. IEEE.

Dinesh T Vasireddy, Dakota S Dale, and Qinghua Li. 2023. Cvss base score prediction using an optimized machine learning scheme. In *2023 Resilience Week (RWS)*, pages 1–6. IEEE.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, et al. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *arXiv preprint arXiv:2412.13663*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Shaochen Zhong, Bing Yin, and Xia Hu. 2024. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *ACM Transactions on Knowledge Discovery from Data*, 18(6):1–32.

Zijing Zhang, Vimal Kumar, Michael Mayo, and Albert Bifet. 2022. Assessing vulnerability from its description. In *International Conference on Ubiquitous Security*, pages 129–143. Springer.

A   Appendix: Reproducibility

This section provides the implementation details for the methods evaluated in Section 4. For DistilBERT-E we directly used the open-source implementation from the authors (Cabral, 2022). Due to the lack of an open-source implementation, we re-implemented CVEDrill based on the pre-trained SecureBERT model available on Hugging Face, 2023. Similarly, we fine-tuned ModernBERT starting from the pre-trained model provided on Hugging Face, 2024a. For the closed-source LLMs we tested the `gpt-4o-2024-11-20` and `o3-mini-2025-01-31` snapshots via OpenAI APIs. For the open-source models, we locally run `Meta-Llama-3.1-70B` (Hugging Face, 2024b) and `deepseek-r1:70b` (DeepSeek-AI, 2025) on our server with vLLM (Kwon et al., 2023) and ollama (Ollama, 2025), respectively. For the Vector Store's text embeddings we used the `all-MiniLM-L6-v2` SBERT model (Hugging Face, 2021).

The rest of the section includes examples of the prompts used in the different strategies described in Section 3 for reproducibility of our results. All the prompts reported below are system messages (falling back to user messages when not supported by the LLM, e.g. DeepSeek-R1) which are then followed by an user message containing the actual description of the CVE to be classified. We also provide the *Response Model* used by Instructor to get structured output data from LLMs. For the sake of brevity we only report examples for one of the eight CVSS metrics, i.e. the Attack Vector. The rest of the prompts and *Reponse Models* can be found in our GitHub repository.

Simple Task Description (STD). The same prompt is used for zero-shot and few-shots setting: the text in bold is only present for the latter.

```
You are an expert cybersecurity analyst from NVD.
Your task is to extract the CVSS v3.1 Attack Vector metric
label for the provided CVE description.

Here are some relevant examples.

CVE description: [...]
LABEL: [...]

...
```

Detailed Task Description (DTD). As for STD, the same prompt is used for zero-shot and few-shots setting: the text in bold is only present for the latter. For the sake of brevity, we omitted the full definitions of the labels, which can be found in the CVSS v3.1 Specification (FIRST, 2023b). A similar prompt is also adopted for CVSS v4.0.

```
You are an expert cybersecurity analyst from NVD.
Your task is to extract the CVSS v3.1 Attack Vector metric
label for the provided CVE description.

The Exploitability metrics reflect the characteristics of
the thing that is vulnerable, [...]

The Attack Vector metric reflects the context by which
vulnerability exploitation is possible [...]

The possible values for the Attack Vector metric are:
- NETWORK: The vulnerable component is bound to the network
stack and the set of possible attackers extends [...]
- ADJACENT_NETWORK: [...]
- LOCAL: [...]
- PHYSICAL: [...]
- DON_KNOW: The information provided is not sufficient to
evaluate the attack vector.

Here are some relevant examples.

CVE description: [...]
LABEL: [...]

...
```

Both STD and DTD approaches share the following *Response Model* expressed as a Pydantic[8] BaseModel, the most widely used data validation library for Python and core part of Instructor.

```
class AttackVector(BaseModel):
    attack_vector: Literal["NETWORK", "ADJACENT_NETWORK",
                           "LOCAL", "PHYSICAL", "DONT_KNOW"
    ] = Field(
        description="The Attack Vector CVSS v3.1 metric.
        ↪ DONT_KNOW is used when the information provided
        ↪ is not sufficient to evaluate the attack
        ↪ vector."
    )
```

Full Vector Prediction (FVP). As commented in Section 3, only for this scenario, all the 8 metrics are concurrently predicted with a single prompt.

```
You are an expert cybersecurity analyst from NVD.
Your task is to extract the eight CVSS v3.1 metrics from
the provided CVE description.
If for any of the metrics the information provided is not
sufficient to answer the question, use the value
"DONT_KNOW".
```

B   Appendix: Impact of Don't Know label

This section examines the impact of the Don't Know (DK) label, focusing on the top-2 LLM configurations for both the full dataset (FD) and low-resource (LR) scenarios from Sec. 4.1 and 4.2: i.e. "24s STD GPT-4o" and "24s STD LLaMA3". The *"DK"* columns in Table 5 show the number and percentage of samples predicted as DK for each CVSS metric. The *"WCL OK"* columns indicate cases where converting a DK label to the Worst Case Label (WCL) matches the Ground Truth. For example, GPT-4o predicted DK for 1% and 1.8% of samples in the full dataset (FD) and low-resource (LR) scenarios, respectively. While the DK sample

Table 5: Analysis of Don't Know (DK) labels: number of cases and impact on performance.

| CVSS | 24s STD GPT-4o (FD) | | 24s STD GPT-4o (LR) | |
Metric	DK	WCL OK	DK	WCL OK
AV	55 (1.0%)	33 (60%)	7 (1.6%)	5 (71%)
AC	40 (0.7%)	33 (82%)	6 (1.4%)	6 (100%)
PR	54 (1.0%)	29 (54%)	5 (1.2%)	1 (20%)
UI	69 (1.3%)	49 (71%)	12 (2.8%)	6 (50%)
S	31 (0.6%)	7 (23%)	5 (1.2%)	4 (80%)
C	60 (1.1%)	23 (38%)	7 (1.6%)	4 (57%)
I	37 (0.7%)	17 (46%)	10 (2.3%)	4 (40%)
A	100 (1.8%)	53 (53%)	9 (2.1%)	5 (56%)
AVG	55 (1.0%)	30 (55%)	7 (1.8%)	4 (57%)

quota is small, using WCL improves performance in over half of these cases for both data availability scenarios. LLaMA3 is omitted from the table because, interestingly, never returned any DK label for all the 8 CVSS metrics in both data scenarios.

C   Appendix: LLM Fine-tuning

This section includes a preliminary evaluation of LLM fine-tuning for GPT-4o via OpenAI APIs (o3-mini does not support fine-tuning yet). Due to high costs, we restricted ourselves to the low-resource setting (fine-tuning on the full dataset would cost 10 times more). We also excluded fine-tuning of the two local open-source models due to our limited resources. Starting from the same train/test sets split of Sec. 4.2, performances are computed on the test set, while the train set is used for LLM fine-tuning. In terms of prompt strategies, we excluded both few-shots prompting (the train set samples are required for fine-tuning itself and cannot be also re-used in the Vector Store) and zero-shot FVP (it's impractical to manually annotate the reasoning chains for the whole train set, cf. Sec. 3), and we selected the best zero-shot approach for GPT-4o, i.e. STD (cf. Table 7). With reference to Table 3 from Sec. 4.2, fine-tuning "0s STD GPT-4o" outperforms ModernBERT on A (92.3%), has the same wF1 (92.0%), but has a worse MF1 (84.2%). In summary, when considering an absolute ranking based on the average of A, wF1, and MF1, ModernBERT still maintains its top-1 position. Due to the limitations described above, we kept LLM fine-tuning out of the scope of the paper and we leave a more comprehensive analysis for future works.

D   Appendix: All zero-shot configurations

Tables 6, 7 and 8 in this section include the entire set of zero-shot methods, grouped by configuration, for the three data availability settings described in

Table 6: CVSS v3.1 full dataset evaluation of all the zero-shot prompting approaches with four LLMs.

Method	A	wF1	MF1	MAE	MSE
0s FVP o3-mini	0.846	0.836	0.722	1.200	3.991
0s FVP GPT-4o	0.754	0.743	0.643	1.655	4.861
0s FVP LLaMA3	0.739	0.733	0.593	1.694	6.178
0s FVP DeepSeek	0.614	0.532	0.352	2.654	10.235
0s STD o3-mini	0.804	0.793	0.676	1.609	6.000
0s STD GPT-4o	0.757	0.765	0.651	1.317	3.343
0s STD DeepSeek	0.741	0.740	0.617	1.430	3.815
0s STD LLaMA3	0.709	0.700	0.561	1.554	5.009
0s DTD o3-mini	0.767	0.755	0.654	2.095	8.747
0s DTD GPT-4o	0.750	0.751	0.634	1.494	4.573
0s DTD DeepSeek	0.736	0.731	0.618	1.597	5.306
0s DTD LLaMA3	0.697	0.676	0.562	2.934	15.788

Table 7: CVSS v3.1 low-resource setting evaluation of all the zero-shot prompting approaches with four LLMs.

Method	A	wF1	MF1	MAE	MSE
0s FVP o3-mini	0.843	0.830	0.722	1.196	3.909
0s FVP GPT-4o	0.747	0.727	0.626	1.756	5.366
0s FVP LLaMA3	0.743	0.735	0.601	1.574	5.308
0s FVP DeepSeek	0.613	0.526	0.344	2.701	10.956
0s STD o3-mini	0.805	0.790	0.669	1.653	6.380
0s STD GPT-4o	0.770	0.772	0.657	1.305	3.294
0s STD DeepSeek	0.741	0.736	0.625	1.401	3.773
0s STD LLaMA3	0.726	0.713	0.595	1.537	4.564
0s DTD o3-mini	0.763	0.749	0.649	2.071	8.552
0s DTD GPT-4o	0.752	0.750	0.631	1.448	4.443
0s DTD DeepSeek	0.741	0.733	0.625	1.391	4.093
0s DTD LLaMA3	0.706	0.682	0.592	2.774	14.687

Sections 4.1, 4.2 and 4.3, respectively. Within each configuration, the entries are sorted in descending order according to the average of A, wF1, and MF1.

E   Appendix: Qualitative Severity Rating Scale (QSRS)

This section evaluates the performance of the methods from Sec.4 in the light of works from Category II (see Sec.2). As done for the MAE/MSE computation for severity scores, we start from the *predicted* 8 CVSS metrics for each sample and *compute* the resulting severity score with the CVSS v3.1 Calculator (NVD, 2024b). This score is then quantized into the 5 levels (None, Low, Medium, High, Critical) defined by the Qualitative CVSS Severity Ratings (QSRS) (FIRST, 2023a). Notice that no model is trained/fine-tuned here: we simply re-evaluate performance as a five-class classification problem. While we still report the same types of metrics used in Tables 2 and 3 in Sec. 4, the results in this section refers to a completely different task, and therefore direct numerical comparisons across tasks are not applicable. Table 9 shows results for the same data availability scenarios as Secs.4.1 and 4.2. In the former (left) the best LLM-based method still falls

Table 8: CVSS v4.0 dataset evaluation of all the zero-shot prompting approaches with four LLMs.

Method	A	wF1	MF1	MAE	MSE
0s FVP o3-mini	0.701	0.684	0.519	1.268	2.763
0s FVP LLaMA3	0.660	0.680	0.523	1.850	6.614
0s FVP GPT-4o	0.629	0.624	0.486	1.379	3.545
0s FVP DeepSeek	0.543	0.442	0.251	2.334	7.648
0s STD o3-mini	0.715	0.710	0.564	1.187	2.596
0s STD DeepSeek	0.627	0.616	0.473	1.092	1.886
0s STD LLaMA3	0.615	0.591	0.460	1.313	3.150
0s STD GPT-4o	0.548	0.539	0.439	1.171	2.371
0s DTD o3-mini	0.780	0.765	0.586	2.103	9.970
0s DTD DeepSeek	0.739	0.727	0.536	1.179	3.309
0s DTD GPT-4o	0.691	0.710	0.547	1.242	2.975
0s DTD LLaMA3	0.689	0.697	0.522	1.984	7.969

Table 9: QSRS Full dataset (left) and low-resource settings (right) evaluation. Top-3 values for each setting are marked in **bold**, ***italic bold*** and *italic*, respectively.

Method	Full dataset			Low-resource		
	A	wF1	MF1	A	wF1	MF1
Worst Case Label	0.173	0.051	0.074	0.211	0.074	0.087
DistilBERT-E	0.718	0.722	0.454	0.677	0.677	0.412
ModernBERT	**0.792**	**0.793**	**0.542**	**0.791**	**0.797**	*0.580*
CVEDrill	**0.792**	*0.792*	*0.534*	0.691	0.696	0.422
24s STD GPT-4o	*0.773*	0.775	*0.515*	*0.770*	*0.776*	**0.580**
24s STD LLaMA3	*0.759*	0.761	0.496	***0.784***	***0.792***	**0.612**

behind CVEDrill and ModernBERT (which, however, swapped their respective positions). When moving to the low-resource case (right), the two few-shots LLMs enter again in the top-3 positions. The Worst Case Label (WCL) baseline performs poorly, reflecting class distribution. With only 17% and 21% of samples labeled *Critical* in the two data scenarios, respectively, assigning this label to all samples yields 17% and 21% Accuracy by design (cf. left and right part of Table 9, respectively).

F   Appendix: Performance breakdown by CVSS metric and *SL+LLM Hybrid* configuration

Tables 10, 11 and 12 break down by individual CVSS metrics the performance for the top-3 methods in the three scenarios from Tables 2, 3, and 4, as detailed in Secs. 4.1, 4.2, and 4.3. The maximum value in each row, for a fixed performance metric, is marked in bold. As discussed in Section 4.2, different configurations may perform better for specific CVSS metrics and considering an hybrid configuration can provide the overall best results. Table 11 highlights in gray the method that maximises the average of A, wF1, and MF1, showing that the optimal "SL+LLM Hybrid" should combine ModernBERT for *UI*, *S*, *C* and *I* CVSS metrics,

Table 10: CVSS v3.1 full dataset evaluation. Performance of top-3 methods, split by CVSS metric. The maximum value in each row for a given performance metric is marked in **bold**.

CVSS Metric	CVEDrill			ModernBERT			24s STD GPT-4o		
	A	wF1	MF1	A	wF1	MF1	A	wF1	MF1
AV	0.938	0.937	0.798	0.941	0.941	0.830	**0.944**	**0.944**	**0.834**
AC	**0.976**	**0.974**	**0.802**	0.975	0.969	0.750	0.965	0.963	0.734
PR	0.851	0.849	0.826	**0.857**	**0.857**	**0.835**	0.854	0.853	0.833
UI	0.957	0.957	0.952	**0.957**	**0.957**	**0.952**	0.933	0.934	0.926
S	**0.976**	**0.976**	**0.962**	0.974	0.974	0.959	0.946	0.947	0.920
C	**0.906**	**0.905**	0.891	0.906	0.905	**0.891**	0.893	0.892	0.874
I	**0.911**	**0.911**	**0.909**	0.901	0.901	0.899	0.887	0.886	0.882
A	**0.921**	**0.917**	0.724	0.914	0.913	**0.745**	0.902	0.902	0.732
AVG	**0.929**	**0.928**	**0.858**	0.928	0.927	0.858	0.915	0.915	0.842

Table 11: CVSS v3.1 low-resource dataset evaluation. Performance of top-3 methods, split by CVSS metric. The maximum value in each row for a given performance metric is marked in **bold**.

CVSS Metric	ModernBERT			24s STD GPT-4o			24s STD LLaMA3		
	A	wF1	MF1	A	wF1	MF1	A	wF1	MF1
AV	**0.942**	0.941	0.874	**0.942**	**0.943**	0.854	0.937	0.938	**0.916**
AC	0.956	0.949	0.732	0.949	0.946	0.737	**0.963**	**0.957**	**0.780**
PR	0.852	0.850	0.783	**0.856**	**0.852**	**0.785**	0.831	0.826	0.746
UI	**0.942**	**0.942**	**0.938**	0.914	0.915	0.909	0.903	0.903	0.897
S	**0.961**	**0.960**	**0.943**	0.940	0.940	0.916	0.905	0.897	0.848
C	**0.884**	**0.883**	**0.872**	0.875	0.875	0.859	0.882	0.882	0.866
I	**0.919**	**0.919**	**0.917**	0.879	0.879	0.873	0.858	0.858	0.850
A	**0.916**	**0.913**	0.725	**0.916**	0.912	**0.730**	0.889	0.886	0.666
AVG	**0.921**	**0.920**	**0.848**	0.909	0.908	0.833	0.896	0.893	0.821

Table 12: CVSS v4.0 evaluation. Performance of top-3 methods, split by CVSS metric. The maximum value in each row for a given performance metric is marked in **bold**.

CVSS Metric	0s DTD o3-mini			0s DTD DeepSeek			0s STD o3-mini		
	A	wF1	MF1	A	wF1	MF1	A	wF1	MF1
AV	**0.947**	**0.947**	**0.962**	0.789	0.787	0.774	0.868	0.865	0.836
AC	**0.974**	0.961	0.493	0.921	0.934	0.479	**0.974**	**0.978**	**0.826**
AT	**0.789**	**0.804**	**0.756**	0.632	0.659	0.604	0.526	0.559	0.504
PR	**0.921**	**0.920**	**0.889**	**0.921**	0.917	0.855	**0.921**	0.918	0.836
UI	**0.868**	**0.859**	0.590	**0.868**	0.856	**0.650**	**0.868**	0.819	0.562
VC	0.500	0.543	0.373	0.553	0.589	0.375	**0.763**	**0.759**	**0.500**
VI	**0.816**	**0.806**	**0.680**	0.789	0.757	0.561	**0.816**	0.781	0.580
VA	0.711	0.684	0.470	**0.763**	**0.763**	**0.522**	0.737	0.727	0.498
SC	**0.605**	**0.553**	0.361	**0.605**	0.551	**0.365**	0.368	0.345	0.263
SI	**0.658**	**0.604**	0.456	0.605	0.530	0.353	0.474	0.485	**0.462**
SA	**0.789**	**0.732**	**0.418**	0.684	0.656	0.360	0.553	0.579	0.338
AVG	**0.780**	**0.765**	**0.586**	0.739	0.727	0.536	0.715	0.710	0.564

Table 13: LLM knowledge cutoff-aware evaluation on full dataset (left) and low-resource settings (right). Top-3 values for each setting are marked in **bold**, *italic bold* and *italic*, respectively.

Method	Full dataset (875)			Low-resource (431)		
	A	wF1	MF1	A	wF1	MF1
Worst Case Label	0.602	0.474	0.278	0.600	0.470	0.278
DistilBERT-E	0.897	0.890	0.729	0.869	0.858	0.673
ModernBERT	**0.921**	**0.919**	**0.839**	**0.921**	**0.920**	**0.848**
CVEDrill	***0.920***	***0.918***	***0.829***	0.887	0.876	0.695
24s STD GPT-4o	*0.909*	*0.908*	*0.827*	***0.909***	***0.908***	***0.833***
24s STD LLaMA3	0.893	0.891	0.798	*0.896*	*0.893*	*0.821*

"24s STD GPT-4o" for *PR* and *A*, and "24s STD LLaMA3" for *AV* and *AC*.

G Appendix: LLM knowledge cutoff-aware additional evaluation

This section provides an additional evaluation where the test data only includes CVEs whose CVSS has been published *after* the LLMs' knowledge cutoff dates: this ensures that the models could not have been possibly exposed to those examples during pre-training. We filtered the results from Tables 2 and 3, retaining only the vulnerabilities whose CVSS publication is after the latest of the LLMs' cutoff dates, i.e. December 2023. After the filtering process, we obtained 875 and 431 CVE test samples for the full dataset and the low-resource scenarios, respectively. For brevity, Table 13 only reports the supervised baselines and LLM few-shot configurations. Notice that the low-resource scenario already includes only samples disclosed in 2024 (cf. Sec. 4.2), thus, by design, all of them have the corresponding CVSS published after December 2023. In other words, the right part of Table 13 is identical to Table 3 and is included here just for the convenience of the reader. From

this table, it is possible to draw conclusions similar to what has been previously observed in Sections 4.1 and 4.2. For the full dataset case (left), the best LLM-based method is still "24s STD GPT-4o" and is still outperformed by CVEDrill and ModernBERT, which however swapped their respective positions in the ranking. For the low-resource case (right), as already discussed in Section 4.2, GPT-4o and LLaMA3 move just behind ModernBERT. This additional evaluation suggests that the performance of LLMs for this prediction task should not be attributed to the memorization of examples the LLM may have been exposed to during the pre-processing phase.