Intelligent Systems with Applications 26 (2025) 200515

Contents lists available at ScienceDirect

Intelligent Systems with Applications

journal homepage: www.journals.elsevier.com/intelligent-systems-with-applications

A comparative analysis on using GPT and BERT for automated vulnerability scoring

Seyedeh Leili Mirtaheri [a,*], Andrea Pugliese [a], Narges Movahed [b], Reza Shahbazian [a]

[a] Department of Informatics, Modeling, Electronics and System Engineering (DIMES), University of Calabria, Rende, 87036, CS, Italy
[b] Mechanical, Energy, and Management Engineering Department, University of Calabria, Rende, 87036, CS, Italy

ARTICLE INFO

Keywords:
Vulnerability scoring
Large language model (LLM)
Transformers
Machine learning

ABSTRACT

Large language models and transformers such as GPT and BERT have shown great improvements in many domains including cybersecurity. A constantly increasing number of vulnerabilities necessitate automated vulnerability scoring systems. Therefore, a deeper understanding of GPT and BERT compatibility with the requirements of the cybersecurity domain seems inevitable for system designers. The BERT model's family is known to be optimized in understanding the contextual relationships with a bidirectional approach, while the GPT models perform unidirectional processing with generative capabilities. Automated vulnerability scoring systems require both the features to analyze the vulnerability and to augment the vulnerability descriptions. On the other hand, powerful GPT models are often more "resource-intensive in comparison with the BERT family. This paper presents a comprehensive comparison analysis of GPT and BERT in terms of their text classification performance, utilizing the vulnerability description classification task. We outline a thorough theoretical and experimental comparison of the models, regarding their architectures, training objectives, and fine-tuning, as well as their text classification performance. We evaluate these models on the vulnerability description classification task and employ rigorous evaluation metrics to shed light on their relative strengths and shortcomings. We also evaluate the hybrid architectures that benefit from combining GPT and BERT at the same time. Our experiment results show that they can effectively leverage the complementary strengths of both GPT and BERT, namely generative and comprehension, leading to further improvements in classification performance.

1. Introduction

The recent improvements of deep learning methods has revolutionized the field of natural language processing (NLP), enabling significant advancements in various domains (Imran & Almusharraf, 2024; Mirtaheri, Greco, & Shahbazian, 2024). Among the most prominent architectures driving these achievements are generative pre-trained transformers (GPT) and bidirectional encoder representations from transformers (BERT). These models, leveraging the *Transformer* architecture, have demonstrated remarkable capabilities in capturing complex patterns and generating human-quality text. These models have received substantial attention in various domains (Crothers, Japkowicz, & Viktor, 2023; Qiu & Jin, 2024; Yang, Luo, Sun, & Luo, 2023). The performance of GPT and BERT models can vary depending on the specific task and the quality of the training data. In some cases, fine-tuning pre-trained models on domain-specific datasets can significantly improve their performance. Moreover, the choice between GPT and BERT may also depend on the computational resources available, as GPT models can be computationally more expensive to train and use compared to BERT.

Vulnerability scoring is a critical task in cybersecurity, where the goal is to assess the potential impact and severity of vulnerabilities discovered in software systems. Security challenges are among the critical issues of ubiquitously utilized software systems, and addressing such security issues demands a thorough approach that involves prioritizing the resolution of potential vulnerabilities by assessing their severity level (Ullah et al., 2018; Zou, Wang, Kolter, & Fredrikson, 2023). Therefore, it is crucial to address vulnerabilities with high risks before attending to those of lesser significance (Lomio, Iannone, De Lucia, Palomba, & Lenarduzzi, 2022; Yitagesu, Zhang, Feng, Li, & Xing, 2021). As a standard approach to evaluate software vulnerabilities, a common vulnerability scoring system (CVSS) introduces a structured approach that utilizes assessment metrics and a severity scoring method to assess the severity of vulnerabilities based on their descriptions (Kühn, Relke, & Reuter, 2023). Critical base metrics, such as Attack Vector,

* Corresponding author.
E-mail address: leili.mirtaheri@dimes.unical.it (S.L. Mirtaheri).

https://doi.org/10.1016/j.iswa.2025.200515
Received 12 December 2024; Received in revised form 12 March 2025; Accepted 28 March 2025
Available online 12 April 2025

Attack Complexity, Authentication, Confidentiality, Integrity, Scope, User Interaction, Privileged Required, and Availability, help experts define a vulnerability and determine how problematic it is. Through a detailed examination of specific vulnerability metrics that outline the unique characteristics of each potential attack, CVSS assigns numerical severity scores, that enhance the overall understanding of the security landscape.

The traditional manual assessment of vulnerability descriptions is a time-consuming process. In particular, the increasing number of reported vulnerabilities and the lack of consistent description formats necessitate automated methods (Elbaz, Rilling, & Morin, 2020). Recently, large language models (LLMs) have gained significant attention due to their generative capabilities, opening up new avenues in various domains. In cybersecurity, LLMs can assist researchers in developing innovative approaches to assess vulnerability severity and mitigate threats (Zhang et al., 2024). Their ability to comprehend and generate human language content offers distinct advantages (Minaee et al., 2024; Zoph et al., 2022), such as enhancing vulnerability descriptions and conducting in-depth analysis (Yao et al., 2024; Yigit, Buchanan, Tehrani, & Maglaras, 2024). LLMs like GPT models have demonstrated proficiency in handling complex cybersecurity tasks, aiding experts in identifying high-risk vulnerabilities and expediting the assessment of their severity. The exploration of LLMs in cybersecurity holds the potential for transformative applications, addressing the challenges posed by the rapidly evolving threat landscape (Yigit et al., 2024).

This paper delves into a comparative analysis of GPT and BERT models, focusing on their architectural differences, applications, and effectiveness in the context of text classification. We present a comprehensive comparison of GPT and BERT in terms of their text classification performance, focusing on automatic vulnerability scoring and classifying software vulnerabilities based on text-based descriptions. To evaluate the performance of GPT and BERT family models in automatic vulnerability scoring, we conduct a series of experiments using publicly available dataset of National Vulnerability Database (NVD). We evaluate the models based on widely adopted metrics such as accuracy, precision, recall, and F1-score. Additionally, we explore the impact of different hyperparameters, pre-training strategies, and fine-tuning techniques on the models' performance.

1.1. Problem statement

Both BERT and GPT models have gained significant recognition for their exceptional ability while they are built on distinct architectures. BERT models use the bidirectional approach to understand contextual relationships while GPTs process unidirectional for generative tasks. In automated vulnerability scoring, we need both comprehension of BERT and the generative potential of GPTs. The first is used to analyze the vulnerability descriptions and the latter one is considered for augmentation of these vulnerability descriptions. It is known that GPT models are powerful but also resource-intensive compared with the BERT models. Therefore, considering the green-friendly applications, it is important to know if the additional computational cost paid by the GPTs is worth the performance. Also, it would be insightful for system designers to know how the combination of BERT and GPT as a hybrid system can improve the gain in compared to the previous methods. Such a comparison is crucial for researchers and practitioners seeking to make informed decisions about which model is best suited for their specific applications in cybersecurity.

For this reason, we present the comparative analysis of the GPT and BERT models' performance in classifying texts. We adopt the application of automatic software vulnerability scoring based on their CVE descriptions (Sun et al., 2023). We seek to shed light on their relative strengths and shortcomings. Additionally, we will explore the potential benefits of combining GPT and BERT in hybrid architectures, leveraging their complementary strengths to achieve even higher performance. We aim to provide valuable insights for researchers and practitioners seeking to apply these powerful language models to their own text classification tasks.

In general, we aim to investigate and answer the following research questions:

- RQ1: Which BERT model from BERT family is the most accurate in the vulnerability descriptions classification task?
- RQ2: Can GPT models be used to identify the severity of vulnerability descriptions?
- RQ3: How do different GPTs impact the performance of vulnerability predictive models, and which GPT model serves the most to enhance the accuracy of these models?
- RQ4: Can GPT models outperform the BERT-based vulnerability predictive models?
- RQ5: Can GPTs be leveraged in creating a hybrid GPT-BERT architecture for vulnerability scoring?

1.2. Contributions

This paper provides a systematic comparative discussion of BERT and GPT models, fulfilling a critical necessity for a refined understanding. The main objective of this research is to transcend the prejudicial comparisons and ambitiously engage with fundamental contrasts that arise between these architectures along multiple dimensions. The paper contributes on the distinct evaluation and extracting BERT and GPT discrimination in a multi-faceted manner. In this respect, we propose a novel comparative framework that assembles a detailed analysis of these models with respect to salient aspects, such as the underlying research questions, architectural designs, considerations for practical deployment, and, most significantly, their distinct mathematical formulations. This essentially provides a much more global perspective whereby both researchers working on BERT or GPT and practitioners-that will use one of these models or the other-will be able to better understand the relative strengths and weaknesses that each model presents in order to make more informed choices for a diverse range of NLP applications. We present a comparative study of text classification-based performance of GPT and BERT models, concentrating the task of automatic vulnerability scoring using software systems' CVE descriptions. We investigate the accuracy performance of both pre-trained and fine-tuned GPT and BERT models on the task of identifying an array of vulnerability metrics, and predicting CVSS vector parameters. Our main contributions are as follows:

- We evaluate and compare the performance of GPT and BERT models in text classification, focusing on the vulnerability descriptions assessments.
- We present a comparative analysis of GPT and BERT models, in terms of their architecture, task-specific performance, and fine-tuning process.
- We demonstrate the effectiveness of integrating GPT models and fine-tuned BERT-based models, in order to develop a hybrid model to automate the vulnerability assessment process.
- We compare different GPTs and BERT models regarding their classification accuracy performance.

The remainder of this paper is organized as follows: Initially, Section 1.1 discusses the problem statement and the research questions we aim to respond to. In particular, Section 1.2 describes our main contributions. In Section 2, we review the related works present in the literature. Section 3 presents a theoretical comparative analysis of BERT and GPT models, regarding their architectures, training objectives, and fine-tuning processes. Following this, Section 4 presents in detail the methodology of our study. Ultimately, the evaluation results are presented and discussed in Section 5 and Section 6 concludes the paper.

2. Related works

GPT-3 represented a significant leap forward, showcasing the potential of LLMs to achieve state-of-the-art performance on a wide range of NLP tasks (Brown, 2020; Ray, 2023). For instance, authors in Gadekallu et al. (2025) investigate the potential of GPT models as supporting tools for higher-education students with learning disabilities. The authors gave a detailed review of how effective GPT usage can be in this area, and then asked themselves how these models can be tailored to give personalized responses to students with disabilities. However, despite their impressive achievements, LLMs still have limitations, such as their reliance on the data they were trained on and their potential for biases.

The field of pre-trained language models (PLMs) has been revolutionized by the introduction of models like BERT (Devlin, 2018). BERT's ability to learn deep, bidirectional representations from unlabeled text has paved the way for state-of-the-art performance on various NLP tasks, including question answering, language inference, and more. However, a major challenge with BERT is its large size and high memory requirements, which has led to research on parameter-efficient models that can achieve comparable performance with fewer parameters. ALBERT (Lan, 2019) is a prime example, demonstrating significant memory reduction while maintaining high performance on benchmark tasks.

2.1. Comparative studies

The authors in Ali, Alajanbi, Yaseen, and Abed (2023) examine the recent developments of AI language models—namely, ChatGPT, Bard, Claude, BERT, and DALL-E—and reflect on how these may be useful in the creative process. The authors consider a variety of possible use cases, demonstrating these tools' encouraging promise to assist in reshaping the terrain of creative enterprises. Zhong et al. Zhong, Ding, Liu, Du, and Tao (2023) conducted a comprehensive comparison between ChatGPT and fine-tuned BERT models on the GLUE benchmark, concentrating on its understanding ability. The findings demonstrate that while ChatGPT demonstrated exceptional performance in inference tasks, it struggled with paraphrase and similarity tasks. The study also highlighted the potential of enhancing ChatGPT's understanding ability through advanced prompting strategies. Relevantly, Zhou et al. Zhou et al. (2024) also provide a detailed overview of Pretrained Foundation Models (PFMs) across various data modalities, focusing on efficiency, security, and privacy. They suggest research directions for enhancing PFMs in areas like scalability, security, and user-friendly interaction. Similarly, the authors in Das, Amini, and Wu (2025) pointed out the challenges of LLMs, focusing on security and privacy of these models. In another study, Topal et al. Topal, Bas, and van Heerden (2021) explored the capabilities of GPT, BERT, and XLNet for Natural Language Generation (NLG) tasks. Their findings emphasized the transformative impact of Transformers in NLG, addressing the limitations of previous RNN and LSTM architectures. Likewise, Yang et al. Yang et al. (2024) offer a practical guide on using LLMs in NLP tasks, discussing model, data, and task perspectives, biases, efficiency, cost, and latency.

The authors in Fields, Chovanec, and Madiraju (2024) conducted a comprehensive survey of text classification techniques, encompassing a wide range of applications and benchmarks. Their study evaluated the accuracy, cost, and safety of LLMs for text classification, challenging the assumption of their universal superiority. Zahid et al. Zahid, Joudar, Albahri, Albahri, Alamoodi, Santamaría, and Alzubaidi (2024) also compared GPT-4 and Google AI (BERT-based architecture) on various capabilities, including translation accuracy, text generation, factuality, creativity, intellect, deception avoidance, sentiment classification, and sarcasm detection. They showed the strengths of GPT-4 in understanding prompts and generating more precise and accurate responses. Qiu et al. Qiu and Jin (2024), on the other hand, investigated the performance of ChatGPT and a fine-tuned BERT model for domain-specific tasks in an intelligent design support system. According to their findings, despite ChatGPT's comparable performance in sentence-level classification, it faces challenges with short sequences. Moreover, ChatGPT's performance improved significantly with a few-shot setting and exhibited knowledge transfer and elicitation capabilities. The comparison between GPT and BERT models even extends beyond text classification. For instance, the authors in De Santis, Martino, Ronci, and Rizzi (2024) explored the use of GPT-4, Mistral, and BERT for user categorization in medical topic posts on Facebook discussion groups. They compared the performance of different pre-processing and embedding techniques, highlighting the strengths of LLMs in achieving optimal user categorization.

GPT and BERT models have also been compared in NLP tasks such as text summarization (Pawar & Makwana, 2022), text generation (Kurup, Narvekar, Sarvaiya, & Shah, 2021), fake news detection (Anirudh, Srikanth, & Shahina, 2024; Horsuwan, Kanwatchara, Vateekul, & Kijsirikul, 2020), emotion recognition (Boitel, Mohasseb, & Haig, 2024), bias detection (Bin Shiha, Atwell, & Abbas, 2023), sentiment analysis (Yadav & Kaushik, 2023), cyberbullying detection (Islam & Rafiq, 2024), and radiology report generation (Hasani et al., 2024) (see Table 1).

2.2. Vulnerability assessment

Automating the process of vulnerability assessment has been performed by the machine learning methods (de Sousa, de Faria, & Miani, 2020; Jacobs, Romanosky, Adjerid, & Baker, 2020; Zhang et al., 2024). These studies often strive to predict CVSS metrics, which have demonstrated potential in severity evaluation using methods such as convolutional neural network (CNN) classifiers, word embeddings (Han, Li, Xing, Liu, & Feng, 2017; Sun et al., 2023), and long short-term memory (LSTM) models (Sahin & Tosun, 2019). A common approach involves analyzing vulnerability descriptions with machine learning and NLP to predict their severity level (Le, Chen, & Babar, 2022; Le, Sabir, & Babar, 2019; Suciu, Nelson, Lyu, Bao, & Dumitraş, 2022). Techniques like support vector machines (SVM) (Almukaynizi et al., 2017; Bozorgi, Saul, Savage, & Voelker, 2010; Sabottke, Suciu, & Dumitraş, 2015), logistic and linear regression (Elbaz et al., 2020; Jiang & Atif, 2020), random forest (Almukaynizi et al., 2019), and neural networks (Sun et al., 2023) are used to identify patterns in descriptions that correlate with exploitability and CVSS metrics (Le et al., 2019; Zhang et al., 2024). Studies have also addressed challenges like new vocabulary in emerging vulnerabilities by incorporating subword features into learning models (Le et al., 2019). Research has extended beyond general severity prediction to focus on specific CVSS metrics, such as privilege required and user interaction metrics (Ognawala, Amato, Pretschner, & Kulkarni, 2018). Social network data has also been leveraged for CVSS prediction using techniques like graph convolutional networks (Chen, Liu, Liu, Park, & Subrahmanian, 2019). Interestingly, some studies propose using a single model for all CVSS metrics, employing the LSTM model and attention mechanisms to share features and improve prediction accuracy across different aspects of vulnerability severity (Gawron, Cheng, & Meinel, 2018; Gong, Xing, Li, Feng, & Han, 2019). Comparing different methods of text embedding like Doc2Vec and TF-IDF, (Kanakogi et al., 2021) suggest Doc2Vec more accurate in detecting assault patterns. Furthermore, deep learning models (Kühn et al., 2023) and neural networks (Guo et al., 2022) show effectiveness in automated vulnerability assessment, hence lowering manual workload and presenting a strong method of classification.

On the other hand, recent research suggests that LLMs have the potential to revolutionize vulnerability assessment (Mirtaheri, Pugliese, Movahed, Majd, et al., 2024; Zhang et al., 2024). These models can be adapted to the cybersecurity domain through continual pre-training on domain-specific data or fine-tuned on labeled data for specific tasks. This opens doors for exploring LLMs like RepairLlama (Silva, Fang, & Monperrus, 2023) and Hackmentor (Zhang et al., 2023), designed specifically for cybersecurity tasks, or even adapting general-purpose

Table 1
Comparing the comparative analyses conducted on large language models.

Paper	Year	Models	Method	Task	Key Findings
Boitel et al. (2024)	2024	GPT, BERT	Emotion recognition	Various datasets	BERT outperforms GPT for emotion recognition.
De Santis et al. (2024)	2024	GPT-4, Mistral, BERT	User categorization	Medical topic posts	LLMs outperform Bag-of-Words based embedding for user categorization.
Anirudh et al. (2024)	2024	GPT-3.5-turbo, BERT	Fake news detection	Tamil news headlines	GPT-3.5-turbo outperforms BERT for fake news detection.
Fields et al. (2024)	2024	Various LLMs, including GPT and BERT	Text classification	Sentiment analysis, question-answering	LLMs are not universally superior; performance varies across tasks.
Zahid et al. (2024)	2024	GPT-4, Google AI (BERT-based)	Capability comparison	Translation, text generation, factuality, creativity, etc.	GPT-4 outperforms Google AI in most capabilities.
Qiu and Jin (2024)	2024	ChatGPT, fine-tuned BERT	Domain-specific tasks	Classification, generation	ChatGPT performs comparably to BERT in classification but struggles with short sequences.
Islam and Rafiq (2024)	2024	GPT-3.5 Turbo, Text-Davinci	Cyberbullying detection	Instagram conversation threads	Both models show promising results for cyberbullying detection.
Hasani et al. (2024)	2024	GPT-4	Radiology report generation	Radiology reports	GPT-4 can generate high-quality radiology reports comparable to human-generated reports.
Bin Shiha et al. (2023)	2023	DistilBert, GPT-3.5-turbo, Google Bard	Bias detection	Higher education news articles	DistilBert shows moderate performance in detecting bias, while GPT-3.5-turbo and Google Bard are unreliable for annotation.
Zhong et al. (2023)	2023	ChatGPT, BERT	Fine-tuning, GLUE benchmark	Understanding ability	ChatGPT outperforms BERT in inference tasks but struggles with paraphrase and similarity.
Yadav and Kaushik (2023)	2023	GPT-2, BERT, DistilBERT	Sentiment analysis	Agricultural domain	Domain-specific BERT outperforms other models for sentiment analysis.
Taneja and Vashishtha (2022)	2022	BERT, DistilBERT	Text classification	Various datasets	Fine-tuned BERT and DistilBERT outperform traditional machine learning approaches.
Pawar and Makwana (2022)	2022	GPT-3, BERT	Text classification	Marathi Polarity Labeled Corpora (MPLC)	GPT-3 outperforms BERT-base for text classification on MPLC.
Kurup et al. (2021)	2021	GPT-2, ELMo, BERT	Text generation	Various datasets	ELMo and BERT outperform word2vec and GloVe for text generation.
Topal et al. (2021)	2021	GPT, BERT, XLNet	NLG	Text generation	Transformers outperform RNNs and LSTMs for NLG.
Horsuwan et al. (2020)	2020	ULMFiT, ELMo, GPT, BERT	Text classification	Thai social media	BERT outperforms other models for Thai social text categorization.
Ali et al. (2023)	2023	ChatGPT, Bard, Claude, BERT, and DALL-E	Creativite Generation	Various Tasks	This new generation of AI models is a key milestone in enhancing our capabilities.
Das et al. (2025)	2025		Review	Security and Privacy Challenges	LLMs are vulnerable to security and privacy attacks, posing risks across various applications, necessitating further research for effective defense mechanisms.
Xu et al. (2024)	2024	Wide range of LLMs	Review	Cybersecurity	Future research should enhance dataset size and diversity, focus on fine-tuning, transfer learning, and domain-specific pre-training, and address interpretability, data privacy, and proactive defense.

models like GPTs through prompt engineering (Zhang et al., 2024). Studies typically adapt pre-trained LLMs for cybersecurity tasks through continual pre-training or supervised fine-tuning (Zhang et al., 2024). Continual pre-training boosts a model's understanding and applicability in cybersecurity by adding more domain-specific, unlabeled data. Supervised fine-tuning, on the other hand, uses labeled data to optimize performance on specific security tasks. This is important for models like ChatGPT, which are designed to follow instructions and stay task-focused. Overall, with the ongoing development of LLMs, this area presents exciting possibilities for future advancements in automated vulnerability assessment.

3. Comparing BERT and GPT models

While both BERT and GPT take advantage of the transformers, they demonstrate distinct characteristics regarding their architecture, training and fine-tuning methodologies, objectives during pre-training, and suitability for various tasks. This section presents a thorough comparison of GPT and BERT models, emphasizing on their principal differences and prospective applications.

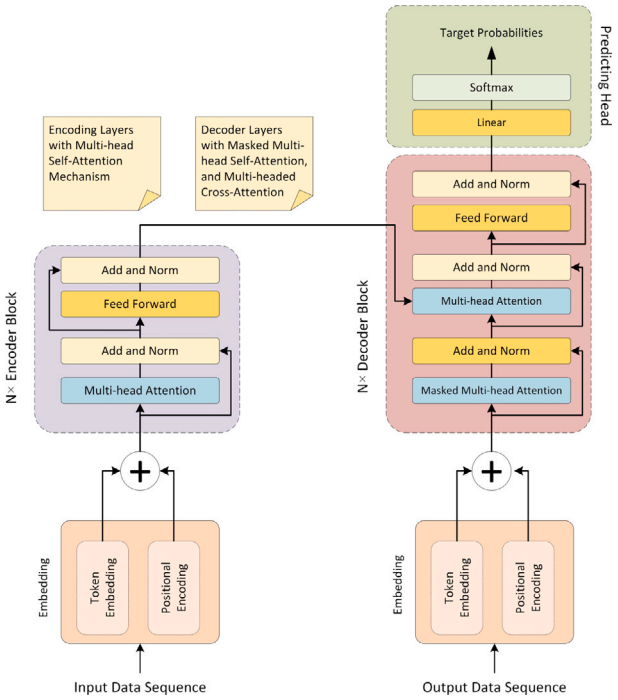

Fig. 1. The general architecture of Transformer presented by Vaswani et al. in 2017.

3.1. Architecture

GPT and BERT models share a common foundation in the architecture, which employs self-attention mechanisms to capture dependencies between words in a sequence. However, they differ in their structural configurations. While GPT is an autoregressive decoder-only model (Pannatier, Courdier, & Fleuret, 2024), designed for generative tasks such as text generation and translation, BERT is an encoder-only model, optimized for understanding and classification tasks. In contrast to GPT models, which process input sequences sequentially, generating output tokens one at a time, BERT models go through input data in both directions simultaneously, capturing contextual information from both preceding and succeeding tokens. This section thoroughly explores the architectures of GPT and BERT models, shedding light on their similarities and distinctions.

3.1.1. Transformer architecture

The core architecture of GPT and BERT models is based on the *Transformer* architecture, introduced by Vaswani (2017), which is illustrated in Fig. 1. The Transformer architecture is a sequence-to-sequence model that eschews the traditional recurrent neural network (RNN) or convolutional neural network (CNN) architectures in favor of a purely attention-based mechanism, which allows the model to capture long-range dependencies in the input sequence more effectively. The Transformer architecture takes advantage of three main components, including *Encoder-Decoder architecture, Self-Attention Mechanism,* and *Positional Encoding*. The encoder processes the input sequence and generates a sequence of hidden representations, while the decoder generates the output sequence, conditioned on the encoder's output.

The Attention Mechanism, being a key component of the Transformer architecture, allows the model to weigh the importance of different parts of the input sequence when computing the representation for a given position, resulting in a context-aware representation for the target word. There are mainly two types of attention utilized in this respect: Scaled Dot-Product Attention and Multi-Head Attention. In comparison to single dot-product attention, while is the most commonly used attention mechanism in GPT models, the latter uses multiple attention heads to capture different aspects of the input sequence. Thirdly, positional encoding is added to the input sequence to provide information about the relative position of words, since the Transformer architecture is position-agnostic. Positional encoding adds a sinusoidal representation of the word's position to the input embeddings, helping the model capture the sequential order of words in the input. Various positional encoding schemes have been proposed, such as sinusoidal positional encoding (Vaswani, 2017) and learned positional embeddings (Radford, 2018).

3.1.2. GPT models

The architecture of GPT models is typically structured using deep stacks of transformer blocks, each consisting of a multi-head attention layer and a fully connected feedforward network. Moreover, residual connections and layer normalizations are used to improve the flow of information through the network, to prevent the vanishing gradient problem, and to help stabilize training. Attention mechanism can enhance the model's ability to capture different types of dependencies, being crucial for understanding complex language patterns and dependencies. This is achieved by performing multiple self-attention operations in parallel, each with a different set of learned parameters. The outputs of these attention operations are then concatenated and projected to a single output dimension.

To understand the evolution and performance of GPT models, it is essential to delve into their underlying architectures. In this regard, the following presents a comparative analysis of different GPT model variants, focusing on their neural network architecture, parameters, and key differences. In overall, the GPT series of models has undergone significant advancements in terms of architecture and performance. From the initial GPT-1 to the latest GPT-4, each stage of development has introduced larger model sizes, improved training techniques, and enhanced capabilities. The following presents the key differences and similarities between these models:

- GPT-1 (Radford, 2018): Introduced in 2018 (Radford, 2018), marked a significant milestone in NLP tasks. It employed a Transformer decoder-only architecture, consisting of 12 layers, each with 12 attention heads. The model was pre-trained on a massive corpus of text data, enabling it to learn contextual representations of words and phrases. GPT-1 model demonstrated impressive results in various NLP tasks, including text generation and question answering.
- GPT-2 (Radford et al., 2019): GPT-2 model was released in 2019 (Radford et al., 2019), proven to be a substantial improvement over its predecessor. It featured a significantly larger model size with 1.5 billion parameters, allowing it to capture more complex language patterns. The architecture remained similar to GPT_1, but with increased depth and width. The model showcased remarkable capabilities in generating coherent and informative text, even for prompts that were relatively short or incomplete.
- GPT-3 (Brown, 2020) and GPT-4: OpenAI continued to advance the GPT series with GPT-3.5 and GPT-4 large language models. These models incorporated refinements to the architecture, including improved attention mechanisms and training techniques. GPT-3.5 demonstrated further improvements in tasks such as code generation and question answering, while GPT-4, released in 2023, is considered the most advanced language model to date. It exhibits enhanced capabilities in various areas, including factual accuracy, creativity, and reasoning (see Fig. 2).

3.1.3. BERT models

BERT models have emerged as a powerful class of language models, demonstrating exceptional performance in a wide range of NLP tasks, particularly suitable for sentiment analysis, question answering, and text classification. Similar to GPT models, BERT's architecture is also

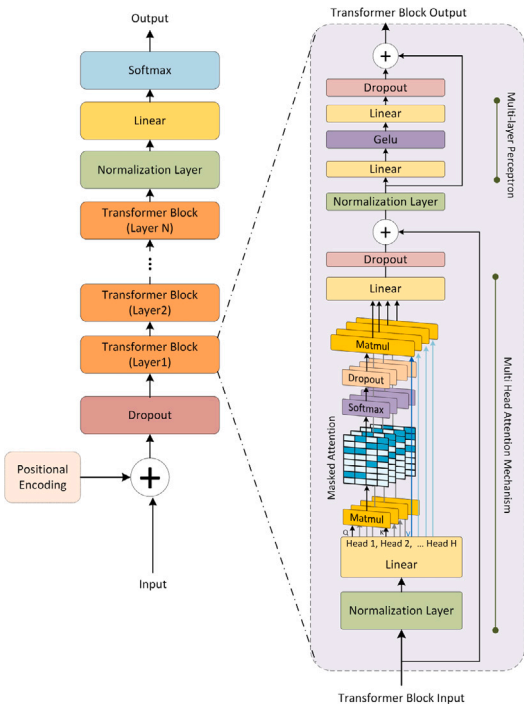

Fig. 2. Overall architecture of the GPT model's foundation.

built upon the Transformer architecture (Vaswani, 2017). However, in contrast to GPTs that utilize a decoder-only transformer architecture, BERT employs an encoder-only transformer architecture. Furthermore, unlike GPTs, which process input data stream sequentially and in one direction, BERT models analyze the complete sequence of words in both directions (Devlin, 2018). This allows them to capture richer semantic information.

Fig. 3 illustrates the overall architecture of BERT_Base Model 3(a), and the architecture of the BERT-based classification model utilized in this study 3(b). BERT model architecture consists of encoder layers, multi-head self-attention mechanisms, and position-wise fully connected feed-forward networks. By encoder layers, the model processes the input sequence and extract relevant information, with bidirectional multi-head attention mechanism in each encoder layer, which enables the model to attend to different parts of the input simultaneously, capturing contextual relationships. On the other hand, the position-wise feed-forward network employed in the model applies non-linear transformations to the input sequence. The BERT model is available in two configurations, namely Base and Large, with the larger version being equipped with enhanced capacity to capture complex language structures. The BERT_Base model consists of 12 encoder layers, with 12 bidirectional self-attention heads on each layer, and a total of 110 million parameters, which makes it more manageable than the BERT_Large model. By contrast, BERT_Large model comprises a total of 340 million parameters, utilizing 24 encoder layers, with 16 attention heads on each layer. Larger models may exhibit superior performance; however, they unavoidably necessitate increased computational resources. Both BERT models are pre-trained on a large corpus of text (BookCorpus and English Wikipedia), providing them with a strong foundation for various NLP tasks.

Numerous BERT model variations have been presented in the literature as a result of the expansion of study in this field.

- **RoBERTa** (Liu, 2019): As a direct successor of BERT, RoBERTa benefits from several engineering advancements while retaining a mostly similar design. Compared to BERT, it has been trained on a somewhat bigger dataset, which has enabled it to understand more intricate linguistic patterns. Furthermore, RoBERTa employs dynamic masking — instead of BERT's fixed masking strategy — where tokens are dynamically masked throughout each training epoch, and undergoes a longer training process, resulting in a more robust model.
- **DistilBERT** (Sanh, 2019): DistilBERT is a more compact and efficient variant of BERT, having been acquired by training a smaller model to imitate the behavior of a bigger model (knowledge distillation (Gou, Yu, Maybank, & Tao, 2021)). DistilBERT functions similarly to BERT but is significantly more efficient, making it suitable for situations with restricted resources.
- **ALBERT** (Lan, 2019): Using a range of methods to lower the number of parameters and increase performance, ALBERT tackles the computational complexity problem related to large-scale BERT models. For example, it uses cross-layer parameter sharing and factorizes the word embedding matrix to minimize the number of parameters. Additionally, it substitutes a sentence-order prediction task — which is more effective and instructive — for the next sentence prediction task.
- **ELECTRA** (Clark, 2020): ELECTRA is a revolutionary pre-training method for language models, using a generative adversarial network (GAN) paradigm, in which a tiny language model generates corrupted versions of the input text, and a larger discriminator model is tasked with identifying the damaged tokens. This training paradigm produces a more robust and efficient model than typical masked language modeling.
- **DeBERTa** (He, Liu, Gao, & Chen, 2020): DeBERTa is an important architectural variant of BERT, introducing disentangled attention, allowing the model to interpret positional and token information independently. This method enhances the model's capacity to detect long-term relationships and comprehend complicated language patterns.

The various BERT models each exhibit unique properties, rendering them suitable for distinct applications. For instance, RoBERTa and ELECTRA demonstrate particular proficiency in tasks demanding a deep understanding of natural language. Conversely, DistilBERT and ALBERT are more efficient options for resource-constrained environments. Moreover, DeBERTa, equipped with a disentangled attention mechanism, can be well-suited for tasks requiring the capture of long-range dependencies.

3.2. Mathematical formulations: BERT VS. GPT

Although BERT and GPT both derive from the Transformer architecture, their mathematical representations and training goals result in much different capabilities and uses. This part will go over the mathematical foundations of these models, stressing their variations and parallels.

3.2.1. Shared mathematical foundation: The transformer core

Both models start by transforming input tokens into dense vector representations using learned token embeddings, then enforced using positional encodings to capture the consecutive character of language. The Transformer layers' starting input representation is obtained by element-wise adding these. The scaled dot-product attention technique is also the basic building block of both models. This method lets the model prioritize various elements of the input sequence while handling every token. Mathematically, for a collection of Values V, Keys K, and Queries Q, attention is computed as:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

where the key vector's dimensionality is d_k and the $softmax$ function guarantees the attention weights total to one. Additionally, both models use multi-head attention to represent complex relationships inside the

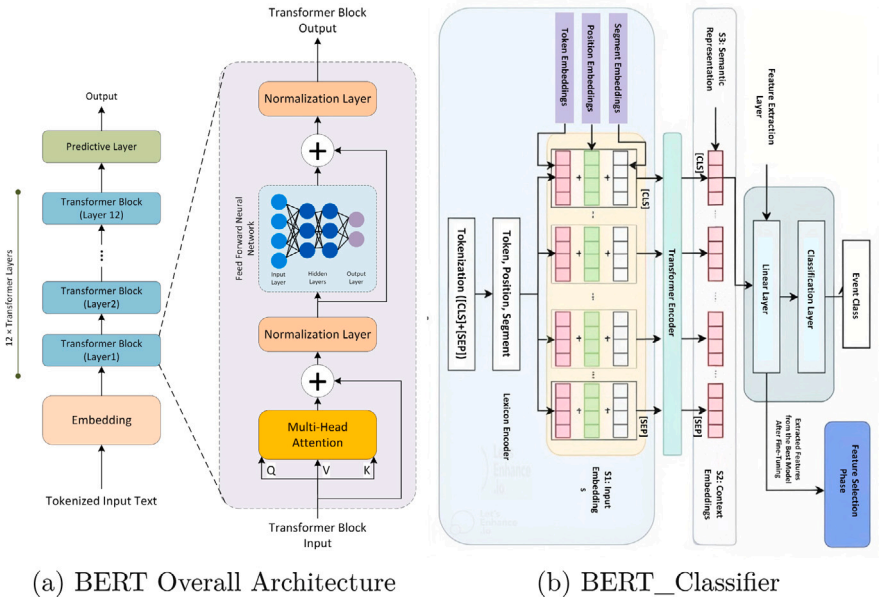

Fig. 3. Overall Architecture of BERT (base) Model (a), and BERT-based Classification Model Architecture (b).

data. This entails running the scaled-dot-product attention mechanism in parallel across several "heads", each having its own learned linear translations for Qs, Ks, and Vs. The final multi-head attention output is obtained by linear transformation of the concatenated output of these heads:

$$MultiHead(Q, K, V) = Concat(head_1, \ldots, head_h)W^O \tag{2}$$

where each for each head we have $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V))$, and (W^O) is a learned projection matrix. Every Transformer layer in both GPT and BERT has a position-wise feed-forward network (FFN) that independently applies to each token's representation. This network adds non-linearity and thereby changes the representations even more:

$$FFN(x) = ReLU(xW_1 + b_1)W_2 + b_2 \tag{3}$$

where W_1, b_1, W_2, and b_2 are learned parameters. Also, both models use residual connections and layer normalization to help with deep network training and guarantee stable gradient flow. Following every sub-layer (attention or FFN), layer normalization is used, and a residual connection appends the input of the sub-layer to its output as $LayerNorm(x + Sublayer(x))$.

3.2.2. Key mathematical distinctions

Although these common structural bases exist, key mathematical variations first come once in the attention masking and training goals, thereby producing fundamentally different model behaviors:

Attention mechanism and masking. In its standard configuration, BERT employs *bidirectional attention* within its Transformer encoder layers. Mathematically, this means that the model considers all other tokens in the input sequence, both preceding and succeeding it, when computing attention for a particular token. The primary design of directionality has no innate masking to hinder the attention span. Although masking is applied during training, it is meant to predict tokens rather than to force directionality. By contrast, GPT utilizes *causal or unidirectional attention*, implemented through an attention mask (M) applied during the scaled dot-product attention calculation:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}} + M\right)V \tag{4}$$

Here, M assigns a large negative value ($\rightarrow -\infty$) to positions corresponding to future tokens. It mathematically enforces that when predicting a token at position i, the model can only attend to tokens at positions ($j \leq i$), effectively preventing it from "looking ahead" to future tokens. This causal masking is essential for autoregressive generation, ensuring the model predicts each token based solely on the preceding context.

Training objectives and loss functions. BERT is pre-trained with the MLM scheme, meaning to randomly masking some input tokens and predicting those tokens based on the bidirectional context of their unmasked counterparts in the sequence. The loss function typically used for MLM is *Cross-Entropy Loss*, wherein the calculation is performed only over the positions of the masked tokens:

$$L_{MLM} = -\sum_{\text{masked positions } i} log(P_{model}(x_i|\text{context})) \tag{5}$$

where the model probability is conditioned on surrounding unmasked context. On the other hand, GPT is pre-trained with *Causal Language Modeling* (CLM), focusing on the prediction of the next token given all the tokens before it. Concretely, given a set of tokens $(x_1, x_2, \ldots, x_n)$, the goal is to maximize the likelihood of predicting each token (x_i) correctly while conditioned upon the whole preceding sequence $(x_1, \ldots, x_{i-1})$. Similarly to BERT, GPT takes as loss function *Cross-Entropy Loss*; however, it is applied to the prediction of each token conditioned on all previous ones in the sequence, as formulated:

$$L_{CLM} = -\sum_{i=1}^{n} log(P_{model}(x_i|x_1, \ldots, x_{i-1})) \tag{6}$$

3.3. Practical training and fine-tuning

While both GPT and BERT models may share a common foundation in using transformer architecture, they differ significantly in their training objectives and applications. In a superficial look, both models undergo a somewhat similar training process, being pre-trained on massive amounts of text data to learn the nuances of language, consequently being able to comprehend and generate human-like language content, respectively. This process includes the acquisition of a comprehensive and varied dataset from multiple sources, followed by the cleaning, tokenizing and pre-processing of the data, and subsequently training the models on this prepared data to predict the missing words in a sequence. In more details, however, GPT is trained on an extensive corpus of textual data to anticipate the subsequent word in a sequence, based on the words that precede it. This unidirectional method renders

it particularly effective for tasks including text generation, translation, and summarization. BERT, in contrast, is trained using a MLM task, in which certain words are replaced with masks and the model must predict the missing terms. Furthermore, BERT undergoes training for a next sentence prediction task, wherein it learns to ascertain whether two sentences are sequential in the context of a given text. This bidirectional methodology allows BERT to capture contextual information from both preceding and subsequent words, thereby enhancing its effectiveness for applications such as question answering, sentiment analysis, and named entity recognition.

More specifically, ChatGPT model is trained through a multi-stage process that leverages the power of deep learning and massive datasets. This process builds upon the foundation laid by its predecessor, *InstructGPT*, and incorporates key refinements to enhance its conversational abilities. InstructGPT was built for following instructions and delivering one-shot responses, whereas ChatGPT is capable of handling multiple inquiries and responses within a conversation, preserving context and consistency. According to OpenAI documents,[1] the ChatGPT model undergoes three distinct training phases, which include generative pre-training, supervised fine-tuning, and reinforcement learning from human feedback. Initially, in the generative pre-training phase, the model is trained on an extensive corpus of text data that covers a wide array of topics and styles. This exposure allows the model to assimilate various linguistic patterns and structures, facilitating its ability to comprehend and generate content that closely resembles human communication.

Subsequently, the model enters the supervised fine-tuning stage. In this phase, the training is conducted using a dataset composed of meticulously curated conversations between human agents. This fine-tuning process aligns the model's responses with human preferences and expectations, while also integrating task-specific knowledge to enhance its conversational capabilities. The final phase of training involves reinforcement learning through human feedback. During this stage, human evaluators assess the quality of the responses generated by ChatGPT. A reward model is then trained to evaluate response quality based on the feedback received. By employing reinforcement learning techniques, this approach ensures that ChatGPT's outputs are in harmony with human values and preferences. Throughout the training process, ChatGPT may encounter several challenges, including misalignment, distributional shift, and over-optimization. Consequently, to mitigate these challenges, the model utilizes techniques such as iterative refinement, diverse datasets, human feedback, and regularization methods.

Notably, a directional data sequence processing approach, such as in GPTs, can inherently limit context leaning. For this reason, BERT models employ MLM and Next Sequence Prediction (NSP) as their pre-training techniques. MLM involves randomly masking a certain percentage of words in each input text sequence and training the model to predict the masked words. To accomplish this, the model uses the context of the data, considering the other non-masked tokens in the input sequence. Consequently, this task encourages the model to learn deep contextual representations that capture the semantic and syntactic relationships between words in a sentence. In more detail, this task requires the model to predict the word as an output vector, converting the predicted vector using a softmax layer, and comparing the predicted vector to the original word's vector using cross-entropy loss. Through the MLM, the model can benefit from contextual understanding, bidirectional representations, and robustness to noise and errors in the input data. However, this may come at a cost of slower convergence.

NSP, on the other hand, involves training the model to predict whether two sentences are consecutive in a given document, assisting the model to learn about sentence-level relationships and coherence. To achieve this, pairs of sentences are created from a corpus of text, with some pairs comprising consecutive sentences from the same document, while others are randomly selected from the corpus. The model is subsequently trained to ascertain whether the given pair of sentences is consecutive. The Next Sentence Prediction task assists the model in comprehending the relationships between sentences and the overall structure of a document, particularly useful for tasks requiring understanding intertextual relations, such as question answering and summarization. Both MLM and NSP techniques play pivotal roles in enhancing the effectiveness of BERT training, being trained together to minimize their combined loss value. MLM facilitates the acquisition of profound contextual representations by requiring the model to predict masked tokens, while NSP fosters an understanding of textual coherence. Through the synergistic combination of these tasks, BERT is capable of learning a robust language model that can be effectively adapted to a wide range of downstream applications.

For optimal performance, the model must undergo fine-tuning specific to the designated task. This process involves adjusting the model's parameters following its training on a dataset tailored to the particular requirements of the assignment (Brown, 2020). GPT models are generally fine-tuned sequentially using supervised fine-tuning, meaning the integration of a task-specific layer upon the pre-trained model, followed by the training of the entire model on a smaller dataset pertinent to the specific task. Furthermore, the fine-tuning process in the GPT model leverages a traversal-style approach to minimize significant alterations to the architecture across various tasks (Radford, 2018). This methodology capitalizes on the knowledge acquired during the pre-training phase to enhance performance in subsequent tasks. The fine-tuning process for BERT models entails integrating the pre-trained model with an additional layer, thereby minimizing the number of parameters that need to be learned from the beginning. Furthermore, one can input task-specific inputs and outputs into BERT and conduct a comprehensive end-to-end fine-tuning of all parameters by training the model on the target dataset (Devlin, 2018).

Table 2 summaries the comparison between GPT and BERT models, regarding different aspects, including architecture, training objective, applications, and strength and potential limitations.

4. Methodology

This section delineates the methodological framework utilized to assess and compare the efficacy of GPT and BERT models within the realm of automated software vulnerability assessment. To achieve this objective, we leverage the NVD vulnerability descriptions dataset and employ these models for text classification tasks, categorizing different vulnerability elements based on their severity. Our methodology is structured in two primary aspects, including (1) **Model Evaluation** (RQ1-3), and (2) **Hybrid Model Exploration** (RQ5). At first, we conduct a comprehensive evaluation of both GPT and BERT models to assess their capabilities in predicting vulnerability elements. This evaluation involves training and testing the models on the NVD dataset and analyzing their performance metrics, such as accuracy, precision, recall, and F1-score. Moreover, we also explore the impact of fine-tuning the models on their accuracy performance. Next, recognizing the potential benefits of combining generative and comprehension tasks, we explore the possibility of developing a hybrid model integrating GPT and BERT. This hybrid approach aims to enhance the automated vulnerability prediction process by leveraging the strengths of both models. By combining GPT's generative capabilities with BERT's comprehension skills, we anticipate achieving more accurate and informative vulnerability assessments. Through our evaluations, we aim to answer the research questions outlined in 1.1.

4.1. Text classification evaluation: Vulnerability assessment

The text classification evaluations of GPT and BERT models carried out for this objective mainly focus on addressing the research questions RQ1 to RQ4. This procedure commences by applying the pre-trained

[1] https://openai.com/index/chatgpt/

Table 2
Comparison of the GPT and BERT models based on their features.

Feature	GPT	BERT
Architecture	Decoder-only transformer	Encoder-only transformer
Directionality	Unidirectional (left to right)	Bidirectional
Training Objective	Unidirectional next word prediction	Masked language modeling and next sentence prediction
Strengths	Strong generative capabilities	Excellent for understanding contextual relationships
Potential Weaknesses	Can be prone to generating repetitive or nonsensical text	May struggle with long-range dependencies in some tasks
Applications	Generative tasks, including: Text generation Machine translation Summarization Creative writing	Diverse range of tasks, including: Question answering Sentiment analysis Named entity recognition Text classification Summarization

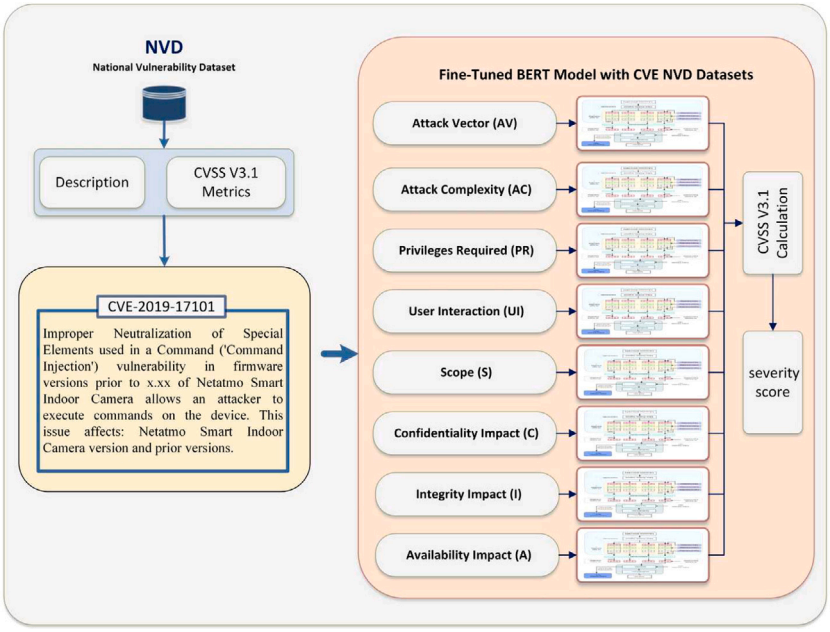

Fig. 4. An overview of the examined BERT based vulnerability predictive architecture.

Bert-base-uncased tokenizer on the augmented data. Subsequently, we utilize different BERT-based models to predict vulnerability metrics. These models are fine-tuned on over 137,500 vulnerability descriptions collected from NVD, excluding a sampled subset of 6370 vulnerability descriptions that are used for evaluation purposes. This fine-tuning process tailors the model to the specific domain of vulnerability descriptions. Furthermore, to effectively capture the nuances associated with different vulnerability metrics, the BERT models are fine-tuned on each metric individually. This approach acknowledges that diverse metrics offer insights into distinct aspects of a vulnerability. By fine-tuning the model for each metric, we enhance its ability to grasp the intricacies present in the descriptions related to that specific metric, ultimately leading to more accurate predictions. Finally, the CVSS severity score evaluator is employed to analyze the vector of various metrics associated with a vulnerability. This analysis ultimately calculates the overall severity score based on the individual metric values. Fig. 4 demonstrates the overall architecture of the explored BERT-based model for vulnerability descriptions classification task.

Similarly, the GPT4 and Fine-tuned GPT3.5turbo models are used to analyze software vulnerability descriptions, and generate their corresponding CVSS vector. The GPT3.5turbo model used in this section is fine-tuned based on 1000 randomly samples of the original descriptions and CVSS vectors, with 800 samples used for training and 200 for the validation purposes. The fine-tuning dataset is created using the original NVD descriptions and the correct CVSS vectors specified in this database, using the following system and prompt messages. The fine-tuning process is carried out using the OpenAI platform, with 10 epochs and batch size of 5. This allowed us to use the OpenAI's managed environment, hence simplifying the process without requiring external computational resources. Finally, the results of GPT-based classifiers are compared to the BERT-based models (RQ5). Fig. 5 illustrates the overall architecture of the GPT-based vulnerability scoring evaluator model. Overall, by inferring the severity score components from the CVE data, this architecture strives to effectively shorten the time gap between identifying vulnerabilities and assigning CVSS scores. The presented evaluation of the two-phased strategy of employing LLMs holds promise to optimize the CVE management procedure and can provide security experts with essential data for prompt and efficient vulnerability mitigation.

4.2. Hybrid architecture

Additionally, we also study the potential of developing a hybrid system, taking advantage of and merging generative and comprehension abilities of GPT and BERT models. Fig. 6 demonstrates the architecture of the hybrid GPT-BERT classification model for an automated

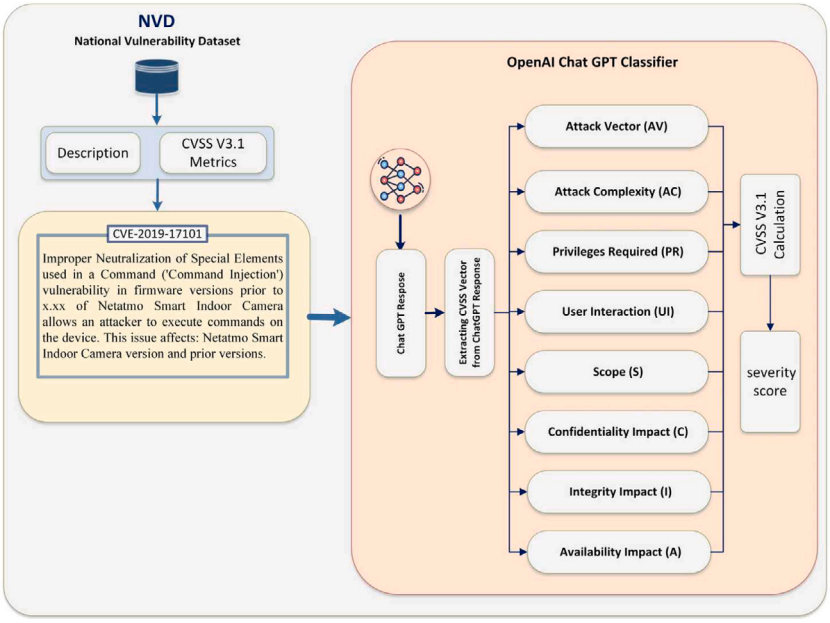

Fig. 5. An overview of the examined GPT based vulnerability predictive architecture.

System Message: "You act as cybersecurity exploitability metrics evaluator.
You are given the CVE description of a software vulnerability. Your task is to evaluate the vulnerability metrics based on the given description, and respond with the CVSS vector associated with the description. The Vulnerability Metrics are as follows:
"Attack Vector" (av), "Attack Complexity" (ac), "Privileges Required" (pr), "User Interaction" (ui), "Scope" (s), "Confidentiality" (c), "Integrity" (I), and "Availability" (a).
First, read the description and understand it fully.
Then, identify the "METRIC VALUE" for each of the "METRICS", in the domain of cybersecurity "vulnerability".
"METRICS" and their range of "VALUES":
"av: [N, L, A, P], ac: [H,L], pr: [N, L, H], ui: [N, R], s: [U, C], c: [H, L, N], i: [H, N, L], a: [H, N, L]"
Your response for each of the METRIC VALUES, MUST be only from the above listed VALUES.
For example, for "av", only the values of [N, L, A, P] are valid, while for the metric "c", the values of [H, L, N] are valid.
Response only in the following format with only one word and do not explain anything:
METRIC: METRIC VALUE."

vulnerability description assessment system. By developing this, we aim to explore the effectiveness of a hybrid GPT-BERT architecture in enhancing the automated vulnerability prediction process. The explored hybrid architecture, entitled FT_GPT_BERT, aims to answer the research question RQ5 and follows a four-step process. First, it leverages the GPT model to generate synthetic data that adheres to the official CVE template, essentially creating more detailed and consistent vulnerability descriptions. Then, a BERT model, specifically fine-tuned on the entire collected national vulnerability database (NVD) dataset, analyses these descriptions for each individual CVSS parameter to predict the corresponding vulnerability metric. Finally, by combining these individual scores using the standard CVSS formula, the model arrives at the final overall CVSS score for the vulnerability.

Prompt: "You are given the CVE description of a software vulnerability. Your task is to evaluate the vulnerability metrics based on the given description, and respond with the CVSS vector associated with the description. First, read the description and understand it fully.
Then, identify the METRIC VALUE for each of the METRICS, in the domain of cybersecurity vulnerability. METRICS and their VALUES:
"av: [N, L, A, P], ac: [H,L], pr: [N, L, H], ui: [N, R], s: [U, C], c: [H, L, N], i: [H, N, L], a: [H, N, L]"
Response only in the following format with only one word and do not explain anything:
METRIC: METRIC VALUE
Description:
"{description}"

To generate the synthetic dataset of vulnerability descriptions in the hybrid architecture, we have examined different GPT-based LLMs, including GPT3.5turbo, Fine-tuned GPT3.5turbo, and GPT4. These models are utilized to refine the collected NVD vulnerability descriptions in alignment with the standard CVE format. To assess the effect of fine-tuning on such models, the GPT3.5turbo LLM undergoes the fine-tuning process using content taken from the original dataset and the manually specified template-based descriptions. To do this, we randomly selected 120 descriptions from the NVD dataset and manually crafted desired reformatted descriptions, adhering to the official CVE format provided in below. The desired descriptions are created manually using these elements and following the pattern and keywords specified in the CVE template, including the problem type, impact, and the cause of the vulnerability.

CVE Template: "PROBLEM TYPE" in "AFFECTED COMPONENTS" causes "IMPACT" when "ATTACK VECTOR"

After fine-tuning the LLM model, the FT_GPT_BERT model is called to revise NVD original vulnerability descriptions following the standard template. To guide the model correctly, the following specific system message and prompt are used. The model operates with a temperature of 1 and a maximum token of 500 for the API calls.

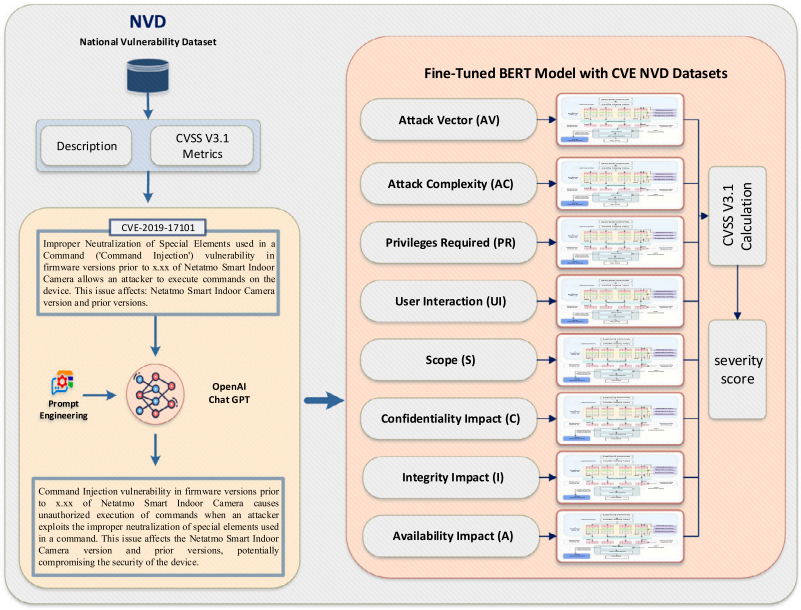

Fig. 6. An overview of the Hybrid GPT-BERT vulnerability classification architecture.

System Message: "You are given the CVE description of a software vulnerability. It would help if you rephrased the Descriptions following this template:
"PROBLEM TYPE" in "AFFECTED COMPONENTS" causes "IMPACT" when "ATTACK VECTOR"
Please generate a comprehensive and formatted description, containing the attack type, impact of the vulnerability, the software component affected, and any attack vectors for the given CVE based on the mentioned template and description."

Prompt: "The CVE description for "{cve_id}" is as follows:
"{description}"
Please generate a comprehensive and reformatted description for this CVE based on the following template. The [DESCRIPTION] field should include an explanation of the attack type, the impact of the vulnerability, the software components affected, and any attack vectors. Do NOT explain and just stick to the given template.
Descriptions often follow this template:
"PROBLEM TYPE" in "AFFECTED COMPONENTS" causes "IMPACT" when "ATTACK VECTOR"

5. Experiments

The experiments conducted in this study are carried out on a 64-bit system with 8 NVIDIA Corporation TU102 [GeForce RTX 2080 Ti] GPUs. The models are trained for 20 epochs, with a batch size of 16, and 0.2 ratio of train and test data. In our experiments, we have used a variety of metrics to give a thorough evaluation. Table 3 describes each metric, and presents their mathematical formulations.

Dataset. The dataset employed in this study was meticulously constructed by extracting over 137,500 software vulnerability descriptions from the NVD website,[2] between 2016 and 2024. To ensure consistency and adherence to established evaluation standards, the dataset was restricted to vulnerabilities following the guidelines outlined in versions 3.0 and 3.1 of the vulnerability assessment framework. Each description encompassed details pertaining to the vulnerability itself, including metrics designated as Attack Vector (AV), Attack Complexity (AC), Privileged Required (PR), User Interaction (UI), Confidentiality (C), D, Integrity (I), Scope (S), Authentication (Au), and Availability (A), along with the descriptions' assigned severity scores.

To outline the subsequent analyses, we randomly sampled a subset of 6370 vulnerability descriptions from the original NVD dataset. This is done while carefully preserving the inherent probability distribution and ratio observed within the original data for each metric and its corresponding labels. The resulting dataset has been thoroughly cleaned and well-balanced, closely resembling the original data's properties, which guarantees reliable and generalizable findings.

5.1. Evaluating BERT classifiers

Here, we present a comprehensive comparison of different BERT predictive models, including BERT, DistilBERT, MediumBERT, SmallBERT, and TinyBERT models on NVD descriptions, on their ability to accurately assess vulnerability descriptions. The models were evaluated across seven different vulnerability elements: AV, AC, PR, UI, C, S, I, and A. The metrics used to assess performance were Accuracy (Acc), Precision (P), Recall (R), F1-score (F1), and Cohen's Kappa Score (Cs). Table 4 illustrates and summarizes the text classification performance of the considered BERT models for different vulnerability elements.

From the Table 4 it is evident that all BERT models achieved reasonably high accuracy rates across most vulnerability elements. However, there were notable variations in performance across different models and vulnerability elements. Despite the mixed results, SmallBERT and MediumBERT models generally showed superior performance on most elements. For instance, SmallBERT and MediumBERT achieved accuracy scores above 0.90 on UI and A, while DistilBERT's accuracy on UI was notably lower at 0.651. As the original model, BERT consistently performed well across most elements, achieving the highest accuracy and Cohen's Kappa Score of 96% and 71% for elements like C and S, respectively. However, its Precision on C was slightly lower at 0.946 compared to MediumBERT's 0.951. On the other hand, the MediumBERT model demonstrated strong overall performance, consistently achieving high accuracy, precision, recall, and F1-scores across multiple vulnerability elements, particularly in assessing AV, AC, and C. This model was able to achieve over 95% accuracy, on AC and C, 95% precision, over 91% recall and F1-score on C, while also showing

[2] https://nvd.nist.gov/

Table 3
Evaluation metrics utilized in our experiments. For Cohen's Kappa Score, P_o and P_e denote the *Observed* and *Expected* agreement, respectively.

Metric	Definition	Mathematical Formulation
Accuracy	Measures the overall correctness of predictions	$\frac{TP+TN}{TP+TN+FP+FN}$
Precision	Reflects the proportion of correctly predicted positive instances out of all predicted positive	$\frac{TP}{TP+FP}$
Recall (Sensitivity)	Measures the proportion of correctly predicted positive instances out of all actual positives.	$\frac{TP}{TP+FN}$
F1-Score	Harmonic mean of precision and recall	$2\times\frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$
Cohen's Kappa Score	Measures agreement between predicted and actual labels, accounting for chance	$\frac{P_o-P_e}{1-P_e}$

superior performance, compared to the other models. In comparison to BERT for element C, MediumBERT showed slightly higher precision and recall values, while matching the highest accuracy. Similarly, the SmallBERT model outperformed the others in classifying UI, S, I, and A, yielding the highest accuracy metrics. This model accomplished over 84% accuracy on S and I, and over 91% on UI and A, while also experiencing minimal gap with the highest accuracy metrics achieved by MediumBERT model. Therefore, it can be seen that the MediumBERT and SmallBERT models showed comparable, superior results, in comparison to larger models such as BERT and DistilBERT, or the smallest model, TinyBERT. In detail, SmallBERT scored 0.841 on S and 0.878 on I, while UI (0.921) and A (0.915) were slightly higher than 0.91. On element UI, compared with MediumBERT, SmallBERT had a slightly better performance score of 0.921 in comparison to 0.906.

Notably, BERT, MediumBERT and SmallBERT models displayed the highest accuracy on all vulnerability elements, indicating their proficiency in identifying the origin of attacks. Among them, however, the SmallBERT model seems to exhibit the highest recall and F1-score values, indicating that this model is effective at identifying positive instances while also maintaining a reasonable balance between precision and recall. For example, SmallBERT was able to achieve the recall of the 0.688 and F1-score 0.716 on AV, and recall of 0.745 and F1-score of 0.790 on AC. Considering the recall of the element AV alone, SmallBERT was able to achieve the highest recall of 0.688, slightly outperforming MediumBERT, which got 0.637. DistilBERT, by contrast, struggled in this area, having the lowest accuracy on UI, S, I, and A. This can be potentially due to its limited capacity to process complex attack scenarios, which may point to the ineffectiveness of distillation, regarding the conducted text classification task. For example, DistilBERT's accuracy was only 0.651 on UI, 0.578 on S, 0.504 on I, and 0.589 on A, significantly lower than the other models. In comparison, SmallBERT achieved accuracy of 0.921, 0.841, 0.878, and 0.915 on UI, S, I, and A, respectively.

Regarding the individual vulnerability elements, AC, AV, and PR, experienced mixed results among different models. While all models performed reasonably well in assessing these elements, with DistillBERT resulting in the highest precision, MediumBERT excelling at accuracy and Cohen's Kappa, and SmallBERT having the recall score. In particular, DistilBERT achieved the highest precision of AV (0.936), AC (0.958), and PR (0.811). MediumBERT achieved the highest accuracy (0.907) and Cohen's Kappa (0.758) on AV and AC (0.951, 0.529) and the highest Cohen's Kappa on PR (0.646). In comparison, SmallBERT had the highest recall on AV (0.688), AC (0.745), and PR (0.716). In particular, the DistillBERT model displayed the highest precision across 5 out of 8 vulnerability elements, namely AV, AC, PR, S, and A, while leading to the lowest accuracy, recall, F1-score, and Cohen's Kappa, on the other elements. Precisely, DistilBERT achieved the highest precision on AV (0.936), AC (0.958), PR (0.811), S (0.859), and A (0.863). Conversely, the findings show the lowest accuracy of around 50% on S, I, and A. For element S, DistilBERT's accuracy of 0.578 was significantly lower than BERT's 0.840, MediumBERT's 0.834, and SmallBERT's 0.841.

Based on the analysis, MediumBERT and SmallBERT emerged as the most effective models for vulnerability description assessment, consistently outperforming both larger and smaller variants across various elements. Their superior performance can be attributed to their ability to process and understand more intricate vulnerability details, or their more accurate fine-tuning. Briefly, on AV and AC, MediumBERT showed excellent accuracy and high Cohen's Kappa scores, while SmallBERT displayed excellent recall on AV, AC, and PR, as well as on accuracy on UI, S, I, and A. While larger models like BERT and DistillBERT may be suitable for certain use cases due to their higher computational requirements, they generally exhibit lower accuracy compared to their medium-sized and smaller counterparts. BERT model performs on average lower than both MediumBERT and SmallBERT across all dimensions. DistilBERT however fares even worse in terms of accuracy, especially relating to the S, I, and A as compared to MediumBERT and SmallBERT. This is noteworthy considering their medium and small model sizes, which may be unexpected given the common assumption that larger and more complex models should yield superior performance. The findings indicate that, at least for the current text classification task, this assertion does not hold true.

RQ1 (*Which BERT model is the most accuracy in vulnerability descriptions classification task?*):
The findings exhibit that the MediumBERT and SmallBERT models demonstrated superior performance in the task of classifying vulnerability descriptions, surpassing larger and smaller models. Larger models, while being computationally demanding, displayed lower accuracy in the examined classification task. This could challenge the conventional assumption that larger models always outperform smaller ones, at least in this specific text classification task.

5.2. Evaluating GPT-based classifiers for vulnerability assessments

The following analysis evaluates the performance of two GPT-based classifiers, GPT4 and a fine-tuned GPT3.5turbo, in assessing vulnerability descriptions. The evaluation focuses on eight vulnerability elements, and based on a range of accuracy metrics.

Table 5 compares the two GPT models regarding the vulnerability description classification. Overall, both GPT models demonstrate strong performance across all vulnerability elements. However, the fine-tuned GPT3.5 turbo, consistently outperforms GPT4 in terms of all accuracy metrics. This clearly suggests that the fine-tuning process has significantly enhanced the model's ability to understand and classify vulnerability descriptions. Noticeably, the fine-tuned GPT3.5turbo model achieved approximately up to 10% improvement in accuracy metric (on AC, C, I). The score of GPT3.5turbo on AC is basically around 0.934 while its score on the same element is viewed as 0.868 for GPT4, showing a robust improvement. Likewise, for the element C, GPT3.5turbo scored with a high accuracy of around 0.838 as opposed

Table 4
Comparing the performance of fine-tuned BERT models including BERT, DistilBERT, MediumBERT, SmallBERT, and TinyBERT on NVD vulnerability descriptions. The classification metrics include Accuracy (Acc), Precision (P), Recall (R), F1-score (F1), and Cohen's Kappa Score (Cs).

	BERT					DistilBERT					MediumBERT				
	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs
AV	0.904	0.793	0.654	0.698	0.738	0.879	**0.936**	0.423	0.427	0.689	**0.907**	0.904	0.637	**0.710**	**0.758**
AC	0.943	0.864	0.722	**0.773**	0.548	0.916	**0.958**	0.500	0.478	0.000	**0.951**	0.929	0.696	0.762	0.529
PR	0.814	0.721	0.684	0.701	0.616	0.761	**0.811**	0.508	0.492	0.468	**0.834**	0.754	0.695	0.720	**0.646**
UI	0.901	0.893	0.894	0.893	0.787	0.651	0.825	0.502	0.399	0.006	0.906	0.905	0.889	0.896	0.793
C	**0.963**	0.946	0.904	0.923	0.847	0.841	0.920	0.500	0.457	0.000	**0.963**	**0.951**	**0.915**	**0.932**	**0.863**
S	0.840	0.837	0.788	0.809	**0.711**	0.578	**0.859**	0.333	0.244	0.000	0.834	0.812	0.803	0.806	0.707
I	0.870	0.875	0.848	0.859	0.787	0.504	0.835	0.333	0.223	0.000	0.852	0.836	0.835	0.836	0.754
A	0.902	0.714	0.621	0.632	0.798	0.589	**0.863**	0.333	0.247	0.000	0.903	0.820	0.708	0.742	0.805

	SmallBERT					TinyBERT				
	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs
AV	0.902	0.768	**0.688**	0.716	0.739	0.878	0.800	0.454	0.465	0.679
AC	0.942	0.863	**0.745**	0.790	**0.581**	0.947	0.846	0.706	0.755	0.512
PR	0.815	0.737	**0.716**	**0.724**	0.616	0.783	0.679	0.616	0.636	0.560
UI	**0.921**	**0.912**	**0.914**	**0.913**	**0.826**	0.887	0.889	0.863	0.873	0.747
C	0.955	0.923	0.908	0.915	0.830	0.957	0.945	0.900	0.920	0.841
S	**0.841**	0.818	0.812	**0.815**	0.718	0.809	0.799	0.757	0.775	0.656
I	**0.878**	**0.882**	**0.865**	**0.873**	**0.797**	0.865	0.864	0.851	0.857	0.782
A	**0.915**	0.768	**0.750**	**0.758**	**0.833**	0.907	0.739	0.635	0.647	0.808

to that of GPT4, which only scored about 0.744. A closer examination of the individual metrics reveals that the GPT3.5turbo classifier consistently achieves higher accuracy, precision, and recall values than GPT4, indicating that it is more effective at correctly identifying vulnerable software descriptions, minimizing false positives, and capturing a larger proportion of actual vulnerabilities. The F1-score, a harmonic mean of precision and recall, further supports the superiority of the GPT3.5turbo model, demonstrating a balanced performance in terms of both classification accuracy and completeness. For AV, the accuracy of the score reached by GPT3.5turbo is 0.869, with precision and recall of 0.868 and 0.869, respectively. While for the same element, GPT4 achieved the score of 0.831, 0.842, and 0.831 for these metrics. Hence, again, for F1-score, AV gets a 0.868 from GPT3.5turbo and 0.835 from GPT4.

Furthermore, the Cohen's Kappa Score provides insights into the agreement between the classifier's predictions and human expert judgments. A higher Kappa score suggests a stronger correlation between the two, indicating that the classifier's predictions are more reliable. In this case, the GPT3.5turbo classifier consistently achieves higher Kappa scores, demonstrating a more robust agreement with human experts than the GPT4 model. To illustrate, GPT3.5turbo has a Cohen's Kappa score of 0.712 on C and 0.782 on S, significantly higher than GPT4's score of 0.488 and 0.059, respectively. While the GPT3.5turbo classifier exhibits superior performance overall, it is important to note that there are slight variations in performance across different vulnerability categories. For instance, both models struggle with *I* and *PR* categories, achieving relatively lower accuracy, recall, and F1-scores. This suggests that these vulnerabilities may pose unique challenges for the classifiers, potentially due to their complexity or ambiguity. For the vulnerability element I and PR, for example, accuracy obtained by GPT4 is around 70% and 80% for GPT3.5turbo, whereas their accuracy for other elements, like AC and S, is considerably higher.

Therefore, the findings presented in this analysis highlight the significant potential of GPT-based models for vulnerability description assessment. The fine-tuning of GPT3.5turbo also leads to substantial improvements in performance compared to the base GPT4 model. However, it is noteworthy to refer to the challenges posed by specific vulnerability categories, potentially requiring additional techniques to enhance the models' capabilities. While GPT3.5turbo predominates over GPT4 in general, the performance scores for the element A have shown a marginal improvement for accuracy at 0.829 versus 0.828, F1-score at 0.828 versus 0.824, and Kappa score at 0.666 versus 0.649 in favor of GPT4. This indicates a marginal exception to the overall trend.

RQ2 (*Can GPT models be used to identify the severity of vulnerability descriptions?*):

Despite being specifically trained for generative purposes, GPTs have shown the potential to analyze the reported vulnerability descriptions, and a good potential to classify descriptions based on their severity levels.

RQ3 (*How do different GPTs impact the performance of vulnerability predictive models, and which GPT model serves the most to enhance the accuracy of these models*):

The findings demonstrate the promising potential of GPT-based predictive models for text classification task, in particular evaluating vulnerability descriptions. While both of the examined GPTs were able to result in comparatively high performance, the Fine-tuned GPT3.5turbo model significantly outperforms the base GPT4 model. This suggests the effectiveness of the fine-tuning process to noticeably enhance the overall performance of GPT models for specific tasks, such as text classification in certain domains. Nevertheless, certain vulnerability categories present challenges that may necessitate further refinements to optimize model performance.

5.3. Comparing the classification of the BERT and GPT models

Here, we conduct a comparative analysis of the performance of the GPT-based classifiers to the BERT-based predictive models. Table 6 illustrates and compares the classification performance of BERT architecture to GPT-based models, in which the results of the model with the highest accuracy value is reported for each vulnerability element.

As it can be seen in Table 6, BERT-based models consistently outperformed GPT-based models across most evaluation metrics and vulnerability elements. For vulnerability attribute C, for example, the BERT-based MediumBERT model displayed a very high accuracy of 0.963, while the accuracy with the GPT-based GPT3.5T_FT model was noticeably lower, with a score of just 0.838. For instance, BERT models generally exhibit superior performance, in terms of precision, suggesting that they were more likely to correctly identify vulnerabilities. To illustrate, the precision of the BERT-based MediumBERT model for AV is 0.904, while the GPT model score for this element is 0.868. This superiority can be attributed to BERT's architecture, which is specifically designed for language understanding tasks and is more adept at capturing contextual relationships within vulnerability descriptions. On the other hand, while showing promising results, GPT-based models

Table 5
Comparing the performance of two GPT-based classifiers on software vulnerability descriptions. The classification metrics are as follows: Accuracy (Acc), Precision (P), Recall (R), F1-score (F1), and Cohen's Kappa Score (Cs).

	GPT4 Classifier					GPT3.5turbo FineTuned Classifier				
	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs
AV	0.831	0.842	0.831	0.835	0.599	**0.869**	**0.868**	**0.869**	**0.868**	**0.663**
AC	0.868	0.878	0.868	0.873	0.134	**0.934**	**0.924**	**0.934**	**0.925**	**0.412**
PR	0.778	0.787	0.778	0.753	0.492	**0.793**	**0.788**	**0.793**	**0.788**	**0.569**
UI	0.817	0.814	0.817	0.815	0.591	**0.856**	**0.856**	**0.856**	**0.855**	**0.679**
C	0.744	0.760	0.744	0.717	0.488	**0.838**	**0.837**	**0.838**	**0.837**	**0.712**
S	0.837	0.856	0.837	0.768	0.059	**0.939**	**0.942**	**0.939**	**0.940**	**0.782**
I	0.718	0.736	0.718	0.692	0.529	**0.808**	**0.817**	**0.808**	**0.806**	**0.682**
A	**0.829**	**0.838**	**0.829**	**0.828**	**0.666**	0.828	0.822	0.828	0.824	0.649

exhibited some limitations, as their performance was generally lower than BERT-based models, especially in terms of accuracy and Cohen's Kappa Score. This might be due to GPT models being specifically designed and trained for generative purposes, which may have influenced the performance of these models regarding the text classification task. For instance, the Cohen's Kappa score for the MediumBERT model for the element AC is 0.529, while the GPT3.5T_FT model scores 0.412. The accuracy on UI also shows consistency alongside that of AC, with SmallBERT achieving 0.921 while GPT3.5T_FT attained 0.856.

In closer look and examining performance across different vulnerability categories, both BERT and GPT models demonstrated varying levels of success. For instance, BERT-based models excelled in vulnerability elements such as UI, C, and I, resulting in the highest accuracy scores across all metrics. More particularly, SmallBERT had an accuracy value of 0.921 for UI, MediumBERT achieved 0.963 for C, SmallBERT reached 0.878 for I, all of which are well above the accuracy figures for those aspects based on GPT models. Above from these elements, these models also achieved the highest accuracy, and Cohen's Kappa score across the majority of elements, including AV, AC, PR, and A, indicating a more reliable overall classification performance as well as a stronger agreement between the model's predictions and human annotations. For instance, on the vulnerability element AV, MediumBERT managed to achieve the highest accuracy of 0.907 and a Cohen's Kappa score of 0.758. On the other hand, with respect to the element AC, the results for MediumBERT were an accuracy of 0.951 and a Cohen's Kappa score of 0.529. Similarly, for PR, MediumBERT got an accuracy of 0.834 and a Cohen's Kappa score of 0.646, and for A, SmallBERT achieved an accuracy of 0.915 and a Cohen's Kappa score of 0.833-all outperforming both metrics of the GPT models on these vulnerability elements. This is while GPT-based models showed competitive accuracy performance on S, also resulting in higher recall and F1-score on elements like AV, AC, PR, and A. Notably, the GPT3.5T_FT turned out to be the superior accuracy of 0.939 for S, compared to 0.841 by SmallBERT. Further, for recall and F1-score on AV, GPT-based model achieved 0.869 and 0.868, surpassing MediumBERT's 0.637 and 0.710. Notably, the findings for F1-score shows mixed results, showing that each of BERT and GPT modes may be superior to the other at some elements, while being worse at other elements. BERT-based models show higher F1 scores regarding the elements UI, C, and I elements, whereas, superior scores are obtained from the models based on GPT for AV, AC, PR, S, and A. Moreover, the two models seem to offer relatively comparative performance, with marginal gaps in accuracy metrics. Therefore, it can be concluded that the choice of model may depend on the specific requirements of the application. For example, if high precision is crucial, a BERT-based model might be preferred, while GPT-based models could be considered for applications where recall is more important. Furthermore, the performance of the models may also ride on the specific domain of text classification. As an example, some vulnerability elements, such as AV, AC, PR, and C experienced higher prediction accuracy with BERT models, whereas the GPT models demonstrated superior accuracy in other elements like S.

RQ4 (*Can GPT models outperform the BERT-based vulnerability predictive models?*):

The optimal model selection may be contingent upon the particular application's demands. For instance, BERT-based models might be favored for applications requiring high precision, while GPT-based models could be considered for those prioritizing recall. Additionally, model performance may be influenced by the specific domain of text classification. For example, BERT models exhibited superior accuracy in predicting certain vulnerability elements (e.g., AV, AC, PR, C), whereas GPT models demonstrated higher accuracy for others (e.g., S).

5.4. Evaluation of the hybrid architecture

This section will evaluate the performance of the explored hybrid architecture that combines the strengths of GPT and BERT models for vulnerability description classification. The hybrid approach leverages GPT to regenerate vulnerability descriptions based on the CVE vulnerability template, thereby enhancing their quality and consistency. Subsequently, BERT models are employed to classify these refined descriptions. By integrating the generative capabilities of GPT with the discriminative power of BERT, the hybrid model aims to achieve superior classification performance compared to either model used independently. Similarly to previous evaluations, the evaluation will focus on assessing the impact of GPT-based refinement on classification accuracy, precision, recall, F1-score, and Cohen's kappa. Additionally, we will investigate the extent to which the hybrid model can mitigate the limitations of individual models and provide a more robust solution for vulnerability description assessment.

Table 8 demonstrates a comparative evaluation of the explored hybrid architecture that combines GPT and BERT models for vulnerability description classification. It illustrates and compares different BERT and GPT models, considering the GPT-BERT architecture, including the original (base) BERT models, and hybrid structures of GPT3.5turbo, fine-tuned GPT3.5turbo, and GPT4 accompanied by different BERT models.

The overall comparison of the experiment results reveals that the hybrid GPT4-BERT model achieves the highest average accuracy across all vulnerability metrics, even outperforming the base BERT model. Considering GPT-based hybrid architecture, it can be seen that the BERT's overall performance increased noticeably compared to the base BERT model. SmallBERT and TinyBERT, on the other hand, exhibited competitive performance with MediumBERT on most metrics for their base models. Similar to the BERT model, these models also experienced an increase in their overall performance on the hybrid GPT-based architecture. Among all the BERT models, however, DistilBERT consistently achieved lower accuracy across all architectures and vulnerability elements.

Even though the base BERT models showed relatively high precision in accuracy metrics, their performance was outperformed by the GPT-integrated hybrid architecture. While the hybrid GPT4-BERT model

Table 6
Comparing the description classification performance of BERT and GPT models.

	BERT-based Models						GPT-based Models					
	Model	Acc	P	R	F1	Cs	Model	Acc	P	R	F1	Cs
AV	MediumBERT	**0.907**	**0.904**	0.637	0.710	**0.758**	GPT3.5T_FT	0.869	0.868	**0.869**	**0.868**	0.663
AC	MediumBERT	**0.951**	**0.929**	0.696	0.762	**0.529**	GPT3.5T_FT	0.934	0.924	**0.934**	**0.925**	0.412
PR	MediumBERT	**0.834**	0.754	0.695	0.720	**0.646**	GPT3.5T_FT	0.793	**0.788**	**0.793**	**0.788**	0.569
UI	SmallBERT	**0.921**	**0.912**	**0.914**	**0.913**	**0.826**	GPT3.5T_FT	0.856	0.856	0.856	0.855	0.679
C	MediumBERT	**0.963**	**0.951**	**0.915**	**0.932**	**0.863**	GPT3.5T_FT	0.838	0.837	0.838	0.837	0.712
S	SmallBERT	0.841	0.818	0.812	0.815	0.718	GPT3.5T_FT	**0.939**	**0.942**	**0.939**	**0.940**	**0.782**
I	SmallBERT	**0.878**	**0.882**	**0.865**	**0.873**	**0.797**	GPT3.5T_FT	0.808	0.817	0.808	0.806	0.682
A	SmallBERT	**0.915**	0.768	0.750	0.758	**0.833**	GPT4	0.829	**0.838**	**0.829**	**0.828**	0.666

achieved the highest overall accuracy, the other GPT models were seen as effective in enhancing the models' accuracy as well. Concerning this, Table 7 provides information regarding the number of vulnerability metrics with the highest accuracy scores, across different models and generated descriptions. As can be seen, the base models yielded the highest precision prediction for merely one vulnerability element *C*, whereas the GPT-based hybrid architectures resulted in comparatively more metrics. Moreover, as these models are utilized to refine the NVD vulnerability descriptions according to the official CVE templates, it can also show the effectiveness of template-based data in enhancing the accuracy of predictive models. Comparing the impact of the variety of GPT-integrated architectures, different BERT models saw mixed results across various generations. While GPT3.5turbo showed more efficacy on the TinyBERT model, it was outperformed by GPT4 and fine-tuned GPT3.5turbo. Notably, when fine-tuned on the template-based descriptions, the GPT3.5turbo was able to enhance the models' performance on different metrics. However, its accuracy score was still outperformed by GPT4 descriptions. Consequently, GPT4 LLM appears to exhibit the most effectiveness in the accuracy of the hybrid GPT-BERT-based models, with the GPT4_BERT model leading the highest accuracy in the prediction of various vulnerability metrics.

Accordingly, the results suggest that BERT models are well-suited and more effective for vulnerability classification tasks with both their base models and GPT-enhanced hybrid. However, the performance can vary depending on the specific BERT model and the vulnerability element being classified. This analysis can also show the effectiveness of a hybrid GPT-BERT architecture, in enhancing the performance of the predictive models, and the potential of GPT-based LLMs utilization in this respect. In this way, the hybrid architecture can leverage the strength of both generative and comprehensive powered large language models, yielding potentially more accurate results in terms of text classification.

RQ5 (*Can GPTs be leveraged in creating a hybrid GPT-BERT architecture for vulnerability scoring?*):

The findings indicate that BERT models, both in their base form and when combined with GPT, are well-suited for vulnerability classification tasks. However, performance may vary depending on the specific BERT model and the vulnerability element being classified. This analysis demonstrates the potential benefits of a hybrid GPT-BERT architecture in enhancing predictive model performance and highlights the utility of GPT-based LLMs in this context. By combining the strengths of generative and comprehensive LLMs, the hybrid architecture can potentially achieve more accurate text classification results.

To expand on our evaluations of the hybrid architecture, and to shed light on the performance and role of BERT classifier, we also assess the accuracy of the GPT classifiers on GPT4-enhanced descriptions. In this respect, a thorough performance assessment of two GPT-based classifiers, GPT4 and the fine-tuned GPT3. 5turbo is given in Table 9.

From the findings, it becomes clear that the fine-tuned GPT-3.5-turbo classifier is evidently superior to the GPT-4 classifier regarding the classification of the GPT4-enhanced descriptions. In all metrics related to vulnerabilities and evaluation metrics, the fine-tuned GPT-3.5-turbo appears to have a greater measure of accuracy and reliability. In particular, regarding Cohen's Kappa values, it seems a strong measurement of agreement with the ground truth on behalf of the fine-tuned model, thus providing classification capability far higher than that of random chance. While GPT-4 does seem to provide a reasonable baseline, the fine-tuned GPT-3.5-turbo presents improvements that suggest task-specific fine-tuning is key for any optimization in vulnerability classification. The greatest differentiation does register for such metrics as AC and S, under which the fine-tuned GPT3.5turbo achieves much higher accuracies.

While both models showed reasonable performance against certain metrics like AV and UI, the overall analysis concludes beyond doubt that such fine-tuning of GPT-3.5-turbo leads to more reliable classification in comparison with zero-shot GPT-4. This underlines the significance of specific training for domain-specific tasks, such as vulnerability assessment. Comparing the two Tables 5 and 9, the performance of both GPT models remained approximately similar, with some minor fluctuations and not much consistent improvement or degradation with enhanced descriptions across different metrics. In the same way, accuracy for PR and S is not very much different, while the models showed some marginal fluctuations, with accuracy increasing for metrics like AV, and falling for A. Importantly, although we can see the effectiveness of the hybrid architecture in enhancing the performance of GPT3.5turbo against GPT4, it fails to provide high accurate classifications when compared to the BERT classifiers.

6. Conclusion and future directions

This study presents a comprehensive comparison analysis of GPT and BERT models, concentrating on their text classification performance, and focusing on the application of automatic software vulnerability scoring. Our presented analysis consists of both technical and experimental evaluations, comparing and differentiating the models with respect to their architectures, training objectives, fine-tuning process, as well as exploratory evaluations. Concerning the vulnerability classification task, we have utilized official NVD vulnerability descriptions, and evaluated different BERT and GPT models, in terms of different individual vulnerability elements.

Our findings demonstrate that both model families are capable of achieving high performance in this task, but their relative strengths and weaknesses vary depending on a number of factors, such as different vulnerability elements or accuracy metrics. On the whole, MediumBERT and SmallBERT, outperformed the other larger and smaller BERT variants, suggesting that model size may not be a direct indicator of performance in all text classification tasks. By contrast, despite being designed for generative purposes, GPT models exhibited promising results in vulnerability description classification. Notably, Fine-tuning GPT models to the certain domain of the input data can significantly enhance their performance for specific tasks. Therefore, it can be concluded that model selection should be determined by the particular requirements of the application. For example, BERT-based models

Table 7
Comparison of number of vulnerability metrics with the highest accuracy scores on original dataset and LLM-based generated ones.

	Original	GPT3.5turbo	Fine-Tuned GPT3.5turbo	GPT4
BERT	1 {C}	3 {AC, PR, A}	1 {S}	4 {AV, UI, I, A}
DistilBERT	0{}	2 {AC, A}	4 {AV, UI, C, S}	2 {PR, I}
MediumBERT	1 {C}	2 {AV, AC}	4 {PR, UI, I, A}	1 {S}
SmallBERT	1 {C}	{}	2 {S, I}	5 {AV, AC, PR, UI, A}
TinyBERT	1 {C}	3 {AV, PR, S}	2 {UI, I}	2 {AC, A}

Table 8
Comparing the performance of fine-tuned BERT models including BERT, DistilBERT, MediumBERT, SmallBERT, and TinyBERT on GPT-based Hybrid Architecture for classifying vulnerability descriptions. The classification metrics include Accuracy (Acc), Precision (P), Recall (R), F1-score (F1), and Cohen's Kappa Score (Cs).

BERT

	Original Base Model					GPT3.5Turbo					GPT3.5Turbo FineTuned					GPT4				
	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs
AV	0.904	0.793	0.654	0.698	0.738	0.947	0.910	**0.870**	0.889	0.865	0.943	0.865	0.823	0.842	0.859	**0.955**	**0.924**	**0.870**	**0.895**	**0.881**
AC	0.943	0.864	0.722	0.773	0.548	**0.968**	0.920	**0.833**	**0.870**	**0.740**	0.957	**0.908**	0.792	0.838	0.677	0.958	0.880	0.810	0.840	0.681
PR	0.814	0.721	0.684	0.701	0.616	**0.895**	**0.853**	**0.852**	**0.852**	**0.795**	0.881	0.836	0.811	0.823	0.762	0.863	0.807	0.788	0.797	0.727
UI	0.901	0.893	0.894	0.893	0.787	0.951	0.956	0.938	0.946	0.892	0.947	0.945	0.941	0.943	0.886	**0.956**	**0.956**	**0.948**	**0.952**	**0.904**
C	**0.963**	**0.946**	**0.904**	**0.923**	**0.847**	0.896	0.894	0.866	0.879	0.814	0.900	0.896	0.867	0.880	0.816	0.878	0.875	0.853	0.863	0.786
S	0.840	0.837	0.788	0.809	0.711	0.969	**0.956**	0.929	0.942	0.884	**0.977**	0.980	0.938	**0.957**	**0.915**	0.976	0.953	**0.960**	**0.957**	0.914
I	0.870	0.875	0.848	0.859	0.787	0.887	0.873	0.887	0.879	0.817	0.903	**0.910**	0.891	0.899	0.843	**0.907**	0.900	**0.901**	**0.900**	**0.847**
A	0.902	0.714	0.621	0.632	0.798	**0.922**	**0.859**	0.721	0.761	0.839	0.902	0.786	0.676	0.703	0.802	**0.922**	0.826	**0.762**	**0.788**	**0.842**

DistilBERT

	Original Base Model					GPT3.5Turbo					GPT3.5Turbo FineTuned					GPT4				
	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs
AV	0.879	**0.936**	0.423	0.427	0.689	0.892	0.805	0.680	0.727	0.720	**0.907**	0.859	**0.752**	**0.796**	**0.764**	0.896	0.733	0.632	0.671	0.717
AC	0.916	**0.958**	0.500	0.478	0.000	**0.940**	0.861	**0.678**	**0.732**	**0.469**	0.932	0.940	0.587	0.630	0.277	0.932	0.921	0.595	0.641	0.297
PR	0.761	**0.811**	0.508	0.492	0.468	0.799	0.752	0.651	0.679	0.580	0.806	0.744	0.693	0.714	0.601	**0.818**	0.758	**0.713**	**0.732**	**0.638**
UI	0.651	0.825	0.502	0.399	0.006	0.885	0.884	0.856	0.868	0.736	**0.905**	**0.896**	**0.902**	**0.899**	**0.797**	0.891	0.886	0.873	0.879	0.757
C	0.841	**0.920**	0.500	0.457	0.000	0.844	0.837	0.809	0.819	0.725	**0.852**	0.836	**0.833**	**0.834**	**0.743**	0.844	0.815	0.829	0.821	0.726
S	0.578	0.859	0.333	0.244	0.000	0.833	0.916	0.500	0.454	0.000	**0.840**	**0.920**	0.500	**0.456**	0.000	0.839	**0.920**	0.500	**0.456**	0.000
I	0.504	0.835	0.333	0.223	0.000	0.856	0.854	0.848	0.851	0.765	0.874	0.874	**0.860**	0.867	0.795	**0.876**	**0.881**	0.859	**0.869**	**0.800**
A	0.589	**0.863**	0.333	0.247	0.000	**0.890**	0.678	**0.680**	**0.678**	**0.773**	0.858	0.853	0.631	0.666	0.711	0.884	0.753	0.639	0.654	0.767

MediumBERT

	Original Base Model					GPT3.5Turbo					GPT3.5Turbo FineTuned					GPT4				
	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs
AV	0.907	0.904	0.637	0.710	0.758	**0.946**	0.875	0.823	0.846	0.864	0.939	0.854	0.840	0.846	0.840	0.945	**0.937**	**0.853**	**0.890**	**0.868**
AC	0.951	0.929	0.696	0.762	0.529	**0.972**	**0.951**	**0.836**	**0.884**	**0.768**	0.948	0.917	0.751	0.809	0.619	0.960	0.886	0.797	0.835	0.670
PR	0.834	0.754	0.695	0.720	0.646	0.854	0.784	0.798	0.790	0.709	**0.889**	**0.862**	0.801	**0.825**	0.771	0.875	0.815	**0.812**	0.814	0.747
UI	0.906	0.905	0.889	0.896	0.793	0.941	0.937	0.935	0.936	0.873	**0.951**	**0.946**	**0.947**	**0.947**	**0.893**	0.941	0.936	0.931	0.933	0.866
C	**0.963**	**0.951**	**0.915**	**0.932**	**0.863**	0.889	0.893	0.857	0.873	0.805	0.870	0.855	0.841	0.848	0.767	0.886	0.882	0.861	0.870	0.802
S	0.834	0.812	0.803	0.806	0.707	0.968	**0.962**	0.935	0.948	0.896	0.969	0.934	**0.951**	0.943	0.885	**0.973**	0.958	**0.951**	**0.955**	**0.909**
I	0.852	0.836	0.835	0.836	0.754	0.892	0.892	0.884	0.888	0.823	**0.906**	**0.901**	**0.897**	**0.899**	**0.845**	0.886	0.886	0.875	0.880	0.813
A	0.903	**0.820**	0.708	**0.742**	0.805	0.917	0.811	0.668	0.699	0.824	**0.918**	0.747	0.713	0.727	**0.833**	0.900	0.757	**0.726**	0.739	0.802

SmallBERT

	Original Base Model					GPT3.5Turbo					GPT3.5Turbo FineTuned					GPT4				
	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs
AV	0.902	0.768	0.688	0.716	0.739	0.943	0.887	0.818	0.846	0.856	0.943	0.885	0.861	0.864	0.847	**0.946**	**0.914**	**0.892**	**0.902**	**0.871**
AC	0.942	0.863	0.745	0.790	0.581	0.963	0.928	0.767	0.826	0.653	0.960	0.892	0.807	0.844	0.687	**0.965**	**0.897**	**0.831**	**0.860**	**0.721**
PR	0.815	0.737	0.716	0.724	0.616	0.858	0.841	0.746	0.781	0.704	0.866	**0.843**	0.773	**0.803**	0.732	**0.879**	0.827	**0.769**	0.794	**0.741**
UI	0.921	0.912	0.914	0.913	0.826	0.929	0.924	0.920	0.922	0.844	0.942	**0.940**	0.929	0.934	0.869	**0.943**	0.936	**0.940**	**0.938**	**0.876**
C	**0.955**	**0.923**	**0.908**	**0.915**	**0.830**	0.884	0.873	0.852	0.862	0.789	0.869	0.859	0.840	0.849	0.767	0.860	0.854	0.830	0.841	0.755
S	0.841	0.818	0.812	0.815	0.718	0.966	0.948	**0.934**	**0.941**	0.881	**0.969**	**0.954**	0.930	**0.941**	**0.883**	0.968	0.951	0.932	**0.941**	**0.883**
I	0.878	0.882	0.865	0.873	0.797	0.892	0.889	**0.883**	**0.886**	**0.823**	**0.893**	0.885	0.879	0.882	0.821	0.888	**0.890**	0.865	0.876	0.814
A	0.915	0.768	0.750	0.758	0.833	0.916	0.759	0.722	0.738	0.831	0.919	**0.887**	0.752	**0.796**	0.836	**0.923**	0.805	**0.768**	0.784	**0.844**

TinyBERT

	Original Base Model					GPT3.5Turbo					GPT3.5Turbo FineTuned					GPT4				
	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs
AV	0.878	0.800	0.454	0.465	0.679	**0.932**	0.854	**0.849**	**0.846**	**0.835**	0.929	0.857	0.817	0.830	0.818	0.921	**0.863**	0.815	0.836	0.810
AC	0.947	0.846	0.706	0.755	0.512	0.958	0.878	0.816	0.843	0.687	0.954	0.832	0.778	0.802	0.604	**0.965**	**0.882**	**0.832**	**0.855**	**0.710**
PR	0.783	0.679	0.616	0.636	0.560	**0.869**	0.635	0.577	0.599	**0.724**	0.848	**0.797**	0.740	0.763	0.684	0.857	0.768	**0.793**	**0.778**	0.710
UI	0.887	0.889	0.863	0.873	0.747	0.936	0.929	0.927	0.928	0.855	**0.946**	**0.942**	**0.937**	**0.939**	**0.879**	0.931	0.922	0.925	0.924	0.848
C	**0.957**	**0.945**	**0.900**	**0.920**	**0.841**	0.890	0.886	0.856	0.870	0.802	0.868	0.854	0.844	0.848	0.767	0.872	0.855	0.843	0.849	0.772
S	0.809	0.799	0.757	0.775	0.656	**0.976**	**0.965**	**0.944**	**0.954**	**0.908**	0.965	0.953	0.916	0.933	0.866	0.974	0.962	0.942	0.952	0.904
I	0.865	0.864	0.851	0.857	0.782	0.892	0.892	0.885	0.888	0.825	**0.898**	0.891	**0.890**	**0.891**	**0.833**	0.886	**0.894**	0.867	0.879	0.813
A	0.907	0.739	0.635	0.647	0.808	0.907	0.808	0.712	0.743	0.809	0.914	**0.900**	0.723	0.771	0.823	**0.927**	0.814	**0.770**	**0.790**	**0.846**

may be favored for applications that demand high precision, whereas GPT-based models might be appropriate for those that emphasize recall.

Moreover, we have also examined the potential of developing a hybrid architecture, leveraging the strength of both family models,

Table 9
Performance Evaluation of GPT Classifiers on GPT-enhanced Descriptions.

	GPT4 Classifier					GPT3.5turbo FineTuned Classifier				
	Acc	P	R	F1	Cs	Acc	P	R	F1	Cs
AV	0.849	0.851	0.849	0.850	0.629	**0.875**	**0.876**	**0.875**	**0.875**	**0.683**
AC	0.870	0.878	0.870	0.874	0.136	**0.928**	**0.914**	**0.928**	**0.917**	**0.341**
PR	0.776	0.781	0.776	0.751	0.486	**0.797**	**0.790**	**0.797**	**0.790**	**0.571**
UI	0.825	0.824	0.825	0.825	0.615	**0.849**	**0.849**	**0.849**	**0.849**	**0.668**
C	0.709	0.731	0.709	0.651	0.392	**0.815**	**0.816**	**0.815**	**0.811**	**0.663**
S	0.833	0.861	0.833	0.759	0.021	**0.929**	**0.931**	**0.929**	**0.930**	**0.749**
I	0.721	0.741	0.721	0.674	0.504	**0.806**	**0.815**	**0.806**	**0.804**	**0.674**
A	0.816	0.808	0.816	0.810	0.624	**0.823**	**0.819**	**0.823**	**0.817**	**0.633**

GPT and BERT, to enhance the overall performance of text classification tasks. In this respect, we have evaluated different variations of the hybrid GPT-BERT architecture, utilizing varied GPT and BERT models. In particular, the GPT models are used to regenerate the NVD descriptions based on the CVE vulnerability template, followed by classification using various BERT models. The evaluation demonstrated the effectiveness of the hybrid architecture, which consistently outperformed individual BERT models across most evaluation metrics and vulnerability elements. Also, the choice of BERT and GPT models within the hybrid architecture was found to significantly impact the accuracy performance, with GPT4-BERT architecture emerging as the promising combination. Experiments demonstrated that this approach achieves significantly higher accuracy (97.7%) compared to the current BERT-based deep learning architectures.

This research lays the groundwork for further advancements in GPT and BERT based text classification models, as well as automated vulnerability assessment. Here, we outline key areas for future exploration. Regarding vulnerability assessment automation, further investigation can be carried out on diverse template structures to streamline vulnerability assessment, which includes developing methods for automatic refinement of vulnerability descriptions. This may significantly reduce the time currently spent on manual evaluation of vulnerabilities. Furthermore, the generated vulnerability assessment descriptions can allow evaluators to prioritize critical issues and expedite the remediation process. Also, by incorporating a broader range of vulnerability parameters and analyzing their interrelationships, one may develop unified models capable of classifying vulnerabilities across various parameter types, eliminating the need for parameter-specific predictive models. This comprehensive approach can significantly improve classification accuracy and provide a holistic automated solution for vulnerability evaluators and software owners. Ultimately, facilitating the efficient handling of newly reported vulnerabilities can substantially reduce overall response times.

Regarding the LLM aspects, the performance of GPT and BERT models can be further enhanced through various architectural and training techniques. One promising approach is to experiment with ensemble methods, such as stacking or bagging, to combine multiple models and potentially improve generalization. Transfer learning can also be employed to fine-tune pre-trained models on domain-specific datasets, potentially leading to better performance and reduced training time. Additionally, investigating techniques to explain the decision-making process of these models can provide valuable insights into their predictions and help identify potential biases. By integrating explainable AI into the large language models, one can significantly enhance the models' trustworthiness and reliability to the users.

On the other hand, evaluating the robustness of GPT and BERT models to adversarial attacks can also help identify vulnerabilities and improve their resilience. Finally, compounding GPT and BERT models into real-world software development and security pipelines can also have a significant impact on these areas. Developing methods for continuous learning will enable the models to adapt to new vulnerabilities and data, ensuring their ongoing effectiveness. Further, incorporating these large language models with other security tools can create more comprehensive and effective solutions for vulnerability assessment and management.

CRediT authorship contribution statement

Seyedeh Leili Mirtaheri: Conceptualization, Methodology, Investigation, Writing – original draft, Software. **Andrea Pugliese:** Supervision, Writing – review & editing, Funding acquisition. **Narges Movahed:** Data curation, Software, Visualization, Writing – review & editing. **Reza Shahbazian:** Methodology, Formal analysis, Visualization, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was partially supported by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union – NextGenerationEU. One of the authors has received financial support from PNRR MUR project FAIR (PE-0000013).

Data availability

I have shared the link of the used data.

References

Ali, A. H., Alajanbi, M., Yaseen, M. G., & Abed, S. A. (2023). Chatgpt4, DALL· e, bard, claude, BERT: Open possibilities. *Babylonian Journal of Machine Learning, 2023*, 17–18.

Almukaynizi, M., Nunes, E., Dharaiya, K., Senguttuvan, M., Shakarian, J., & Shakarian, P. (2017). Proactive identification of exploits in the wild through vulnerability mentions online. In *2017 international conference on cyber conflict* (pp. 82–88). IEEE.

Almukaynizi, M., Nunes, E., Dharaiya, K., Senguttuvan, M., Shakarian, J., & Shakarian, P. (2019). Patch before exploited: An approach to identify targeted software vulnerabilities. *AI in Cybersecurity*, 81–113.

Anirudh, K., Srikanth, M., & Shahina, A. (2024). Multilingual fake news detection in low-resource languages: A comparative study using BERT and GPT-3.5. In B. R. Chakravarthi, B. B, M. A. García Cumbreras, S. M. Jiménez Zafra, M. Subramanian, K. Shanmugavadivel, & P. Nakov (Eds.), *Speech and language technologies for low-resource languages* (pp. 387–397). Cham: Springer Nature Switzerland.

Bin Shiha, R., Atwell, E., & Abbas, N. (2023). Detecting bias in university news articles: A comparative study using BERT, GPT-3.5 and Google bard annotations. In M. Bramer, & F. Stahl (Eds.), *Artificial intelligence XL* (pp. 487–492). Cham: Springer Nature Switzerland.

Boitel, E., Mohasseb, A., & Haig, E. (2024). A comparative analysis of GPT-3 and BERT models for text-based emotion recognition: Performance, efficiency, and robustness. In N. Naik, P. Jenkins, P. Grace, L. Yang, & S. Prajapat (Eds.), *Advances in computational intelligence systems* (pp. 567–579). Cham: Springer Nature Switzerland.

Bozorgi, M., Saul, L. K., Savage, S., & Voelker, G. M. (2010). Beyond heuristics: learning to classify vulnerabilities and predict exploits. In *Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 105–114).

Brown, T. B. (2020). Language models are few-shot learners. arXiv preprint arXiv: 2005.14165.

Chen, H., Liu, J., Liu, R., Park, N., & Subrahmanian, V. (2019). VEST: A system for vulnerability exploit scoring & timing. In *IJCAI* (pp. 6503–6505).

Clark, K. (2020). Electra: Pre-training text encoders as discriminators rather than generators. arXiv preprint arXiv:2003.10555.

Crothers, E. N., Japkowicz, N., & Viktor, H. L. (2023). Machine-generated text: A comprehensive survey of threat models and detection methods. *IEEE Access*, *11*, 70977–71002.

Das, B. C., Amini, M. H., & Wu, Y. (2025). Security and privacy challenges of large language models: A survey. *ACM Computing Surveys*, *57*(6), 1–39.

De Santis, E., Martino, A., Ronci, F., & Rizzi, A. (2024). From bag-of-words to transformers: A comparative study for text classification in healthcare discussions in social media. *IEEE Transactions on Emerging Topics in Computational Intelligence*.

de Sousa, D. A., de Faria, E. R., & Miani, R. S. (2020). Evaluating the performance of Twitter-based exploit detectors. arXiv preprint arXiv:2011.03113.

Devlin, J. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Elbaz, C., Rilling, L., & Morin, C. (2020). Fighting N-day vulnerabilities with automated CVSS vector prediction at disclosure. In *Proceedings of the 15th international conference on availability, reliability and security* (pp. 1–10).

Fields, J., Chovanec, K., & Madiraju, P. (2024). A survey of text classification with transformers: How wide? How large? how long? How accurate? How expensive? How safe? *IEEE Access*, *12*, 6518–6531. http://dx.doi.org/10.1109/ACCESS.2024.3349952.

Gadekallu, T. R., Yenduri, G., Kaluri, R., Rajput, D. S., Lakshmanna, K., Fang, K., et al. (2025). The role of GPT in promoting inclusive higher education for people with various learning disabilities: a review. *PeerJ Computer Science*, *11*, Article e2400.

Gawron, M., Cheng, F., & Meinel, C. (2018). Automatic vulnerability classification using machine learning. In *Risks and security of internet and systems: 12th international conference, cRiSIS 2017, dinard, France, September 19–21, 2017, revised selected papers 12* (pp. 3–17). Springer.

Gong, X., Xing, Z., Li, X., Feng, Z., & Han, Z. (2019). Joint prediction of multiple vulnerability characteristics through multi-task learning. In *2019 24th international conference on engineering of complex computer systems* (pp. 31–40). IEEE.

Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). Knowledge distillation: A survey. *International Journal of Computer Vision*, *129*(6), 1789–1819.

Guo, H., Chen, S., Xing, Z., Li, X., Bai, Y., & Sun, J. (2022). Detecting and augmenting missing key aspects in vulnerability descriptions. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *31*(3), 1–27.

Han, Z., Li, X., Xing, Z., Liu, H., & Feng, Z. (2017). Learning to predict severity of software vulnerability using only vulnerability description. In *2017 IEEE international conference on software maintenance and evolution* (pp. 125–136). IEEE.

Hasani, A. M., Singh, S., Zahergivar, A., Ryan, B., Nethala, D., Bravomontenegro, G., et al. (2024). Evaluating the performance of generative pre-trained transformer-4 (GPT-4) in standardizing radiology reports. *European Radiology*, *34*(6), 3566–3574.

He, P., Liu, X., Gao, J., & Chen, W. (2020). Deberta: Decoding-enhanced bert with disentangled attention. arXiv preprint arXiv:2006.03654.

Horsuwan, T., Kanwatchara, K., Vateekul, P., & Kijsirikul, B. (2020). A comparative study of pretrained language models on thai social text categorization. In N. T. Nguyen, K. Jearanaitanakij, A. Selamat, B. Trawiński, & S. Chittayasothorn (Eds.), *Intelligent information and database systems* (pp. 63–75). Cham: Springer International Publishing.

Imran, M., & Almusharraf, N. (2024). Google gemini as a next generation AI educational tool: a review of emerging educational technology. *Smart Learning Environments*, *11*(1), 22.

Islam, M. S., & Rafiq, R. I. (2024). Comparative analysis of GPT models for detecting cyberbullying in social media platforms threads. In J. A. Lossio-Ventura, E. Ceh-Varela, G. Vargas-Solar, R. Marcacini, C. Tadonki, H. Calvo, & H. Alatrista-Salas (Eds.), *Information management and big data* (pp. 331–346). Cham: Springer Nature Switzerland.

Jacobs, J., Romanosky, S., Adjerid, I., & Baker, W. (2020). Improving vulnerability remediation through better exploit prediction. *Journal of Cybersecurity*, *6*(1), tyaa015.

Jiang, Y., & Atif, Y. (2020). An approach to discover and assess vulnerability severity automatically in cyber-physical systems. In *13th international conference on security of information and networks* (pp. 1–8).

Kanakogi, K., Washizaki, H., Fukazawa, Y., Ogata, S., Okubo, T., Kato, T., et al. (2021). Tracing capec attack patterns from cve vulnerability information using natural language processing technique.

Kühn, P., Relke, D. N., & Reuter, C. (2023). Common vulnerability scoring system prediction based on open source intelligence information sources. *Computers & Security*, *131*, Article 103286.

Kurup, L., Narvekar, M., Sarvaiya, R., & Shah, A. (2021). Evolution of neural text generation: Comparative analysis. In S. K. Bhatia, S. Tiwari, S. Ruidan, M. C. Trivedi, & K. K. Mishra (Eds.), *Advances in computer, communication and computational sciences* (pp. 795–804). Singapore: Springer Singapore.

Lan, Z. (2019). Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942.

Le, T. H., Chen, H., & Babar, M. A. (2022). A survey on data-driven software vulnerability assessment and prioritization. *ACM Computing Surveys*, *55*(5), 1–39.

Le, T. H. M., Sabir, B., & Babar, M. A. (2019). Automated software vulnerability assessment with concept drift. In *2019 IEEE/ACM 16th international conference on mining software repositories* (pp. 371–382). IEEE.

Liu, Y. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

Lomio, F., Iannone, E., De Lucia, A., Palomba, F., & Lenarduzzi, V. (2022). Just-in-time software vulnerability detection: Are we there yet? *Journal of Systems and Software*, *188*, Article 111283.

Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X., et al. (2024). Large language models: A survey. arXiv preprint arXiv:2402.06196.

Mirtaheri, S. L., Greco, S., & Shahbazian, R. (2024). A self-attention TCN-based model for suicidal ideation detection from social media posts. *Expert Systems with Applications*, *255*, Article 124855.

Mirtaheri, S. L., Pugliese, A., Movahed, N., Majd, A., et al. (2024). Advanced automated vulnerability scoring: Improving performance with a fine-tuned BERT-CNN model. In *International symposium on telecommunication*.

Ognawala, S., Amato, R. N., Pretschner, A., & Kulkarni, P. (2018). Automatically assessing vulnerabilities discovered by compositional analysis. In *Proceedings of the 1st international workshop on machine learning and software engineering in symbiosis* (pp. 16–25).

Pannatier, A., Courdier, E., & Fleuret, F. (2024). σ-GPTs: A new approach to autoregressive models. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 143–159). Springer.

Pawar, C. S., & Makwana, A. (2022). Comparison of BERT-base and GPT-3 for marathi text classification. In P. K. Singh, S. T. Wierzchoń, J. K. Chhabra, & S. Tanwar (Eds.), *Futuristic trends in networks and computing technologies* (pp. 563–574). Singapore: Springer Nature Singapore.

Qiu, Y., & Jin, Y. (2024). ChatGPT and finetuned BERT: A comparative study for developing intelligent design support systems. *Intelligent Systems with Applications*, *21*, Article 200308. http://dx.doi.org/10.1016/j.iswa.2023.200308.

Radford, A. (2018). Improving language understanding by generative pre-training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, *1*(8), 9.

Ray, P. P. (2023). ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems*, *3*, 121–154.

Sabottke, C., Suciu, O., & Dumitraş, T. (2015). Vulnerability disclosure in the age of social media: Exploiting twitter for predicting {Real – World} exploits. In *24th USeNIX security symposium (USeNIX security 15)* (pp. 1041–1056).

Sahin, S. E., & Tosun, A. (2019). A conceptual replication on predicting the severity of software vulnerabilities. In *Proceedings of the 23rd international conference on evaluation and assessment in software engineering* (pp. 244–250).

Sanh, V. (2019). DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108.

Silva, A., Fang, S., & Monperrus, M. (2023). RepairLLaMA: Efficient representations and fine-tuned adapters for program repair. arXiv preprint arXiv:2312.15698.

Suciu, O., Nelson, C., Lyu, Z., Bao, T., & Dumitraş, T. (2022). Expected exploitability: Predicting the development of functional vulnerability exploits. In *31st USeNIX security symposium (USeNIX security 22)* (pp. 377–394).

Sun, X., Ye, Z., Bo, L., Wu, X., Wei, Y., Zhang, T., et al. (2023). Automatic software vulnerability assessment by extracting vulnerability elements. *Journal of Systems and Software*, *204*, Article 111790.

Taneja, K., & Vashishtha, J. (2022). Comparison of transfer learning and traditional machine learning approach for text classification. In *2022 9th international conference on computing for sustainable global development (INDIACom)* (pp. 195–200). http://dx.doi.org/10.23919/INDIACom54597.2022.9763279.

Topal, M. O., Bas, A., & van Heerden, I. (2021). Exploring transformers in natural language generation: Gpt, bert, and xlnet. arXiv preprint arXiv:2102.08036.

Ullah, F., Edwards, M., Ramdhany, R., Chitchyan, R., Babar, M. A., & Rashid, A. (2018). Data exfiltration: A review of external attack vectors and countermeasures. *Journal of Network and Computer Applications*, *101*, 18–54.

Vaswani, A. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*.

Xu, H., Wang, S., Li, N., Wang, K., Zhao, Y., Chen, K., et al. (2024). Large language models for cyber security: A systematic literature review. arXiv preprint arXiv:2405.04760.

Yadav, S., & Kaushik, A. (2023). Comparative study of pre-trained language models for text classification in smart agriculture domain. In S. Das, S. Saha, C. A. Coello Coello, & J. C. Bansal (Eds.), *Advances in data-driven computing and intelligent systems* (pp. 267–279). Singapore: Springer Nature Singapore.

Yang, J., Jin, H., Tang, R., Han, X., Feng, Q., Jiang, H., et al. (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.

Yang, B., Luo, X., Sun, K., & Luo, M. Y. (2023). Recent progress on text summarisation based on BERT and GPT. In Z. Jin, Y. Jiang, R. A. Buchmann, Y. Bi, A.-M. Ghiran, & W. Ma (Eds.), *Knowledge science, engineering and management* (pp. 225–241). Cham: Springer Nature Switzerland.

Yao, Y., Duan, J., Xu, K., Cai, Y., Sun, Z., & Zhang, Y. (2024). A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, Article 100211.

Yigit, Y., Buchanan, W. J., Tehrani, M. G., & Maglaras, L. (2024). Review of generative AI methods in cybersecurity. arXiv preprint arXiv:2403.08701.

Yitagesu, S., Zhang, X., Feng, Z., Li, X., & Xing, Z. (2021). Automatic part-of-speech tagging for security vulnerability descriptions. In *2021 IEEE/ACM 18th international conference on mining software repositories* (pp. 29–40). IEEE.

Zahid, I. A., Joudar, S. S., Albahri, A., Albahri, O., Alamoodi, A., Santamaría, J., et al. (2024). Unmasking large language models by means of OpenAI GPT-4 and Google AI: A deep instruction-based analysis. *Intelligent Systems with Applications*, *23*, Article 200431. http://dx.doi.org/10.1016/j.iswa.2024.200431.

Zhang, J., Bu, H., Wen, H., Chen, Y., Li, L., & Zhu, H. (2024). When llms meet cybersecurity: A systematic literature review. arXiv preprint arXiv:2405.03644.

Zhang, J., Wen, H., Deng, L., Xin, M., Li, Z., Li, L., et al. (2023). HackMentor: Fine-tuning large language models for cybersecurity. In *2023 IEEE 22nd international conference on trust, security and privacy in computing and communications (trustCom)* (pp. 452–461). IEEE.

Zhong, Q., Ding, L., Liu, J., Du, B., & Tao, D. (2023). Can chatgpt understand too? a comparative study on chatgpt and fine-tuned bert. arXiv preprint arXiv:2302.10198.

Zhou, C., Li, Q., Li, C., Yu, J., Liu, Y., Wang, G., et al. (2024). A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *International Journal of Machine Learning and Cybernetics*, 1–65.

Zoph, B., Raffel, C., Schuurmans, D., Yogatama, D., Zhou, D., Metzler, D., et al. (2022). Emergent abilities of large language models. *TMLR*.

Zou, A., Wang, Z., Kolter, J. Z., & Fredrikson, M. (2023). Universal and transferable adversarial attacks on aligned language models. arXiv preprint arXiv:2307.15043.