Evaluating and Securing Power Systems against Vulnerabilities Introduced by Large Language Models

Manpo Li[1*], Xuerui Yang[1], Xiaochen Yang[1], Shugui Zhang[2]
[1]Northeast Branch of State Grid Corporation of China, Shenyang 110180, Liaoning, China
[2]Sichuan Energy Internet Research InstituteTsinghua University, Chengdu 610218, Sichuan, China
E-mail: limanponesgcc@163.com, yangxueruin@ne.sgcc.com.cn, philipsyxc@163.com, aix@zhutkj.cn
*Corresponding author

Keywords: Power systems, large language models, security threats, TinyLlama Chat 1.1, HAI Dataset

Received: June 6, 2025

An exciting new direction for improving operational efficiency and decision-making is the use of Large Language Modeling (LLMs) to contemporary power systems. Nevertheless, there may be unanticipated security risks associated with this move. Using LLMs to power networks may pose certain risks, which this paper examines. It stresses the need of doing research and developing remedies immediately. It is a challenging but vital job to secure large language models in a power monitoring context. Through the implementation of thorough security measures, the promotion of a security-conscious culture, and the continuous monitoring of new threats while technologies, we may maximize the benefits of LLMs while minimizing their hazards. It is our duty as information security experts to pioneer this new field and make sure that our security protocols adapt to the increasing sophistication of our AI systems. Security flaws in LLM that allow rapid injection attacks are among the most critical ones. These types of attacks take advantage of LLMs' fundamental features by deliberately feeding them data that will cause them to operate in an unexpected way or leak private information. Many LLMs have seen extensive usage with the introduction of commercially accessible systems like ChatGPT. One crucial area of cybersecurity that is seeing a rise in the use of LLMs is power monitoring systems. It is critical to safeguard these systems against cyber-attacks since they are essential for the stability of society and the nation's energy supply. In order to keep these systems resilient and reliable, it is essential to detect unexpected vulnerabilities, especially zero-day attacks. One potential way to improve these detection capabilities is via LLMs. It integrates power-system standards and threat intelligence with traditional anomaly detection, LLM-assisted reasoning over code/configs/logs, and protocol-aware telemetry. In order to uncover undisclosed power monitoring system weaknesses, we used models with LSTM with GRU. Since these models are masters of sequential data analysis, they are ideal for this job. Sequential data includes things like sensor readings, system logs, and network traffic. By identifying unusual activity that differs from typical system operation, LSTMs and GRUs are able to discover new, "zero-day" vulnerabilities, in contrast to conventional security technologies that depend on predetermined attack signatures. Here, we built an LLM model using TinyLlama Chat 1.1. The LLM takes the processed packet data and the extracted context as input and outputs a user-friendly summary of the packet file. Through the use of machine learning models, the program provides a concise, well-organized, and straightforward overview of the network's operations.

Povzetek: Prispevek obravnava uporabo LLM-jev za nadzor elektroenergetskih sistemov ter poudarja varnostna tveganja in potrebo po naprednih metodah zaznavanja napadov za zanesljivo delovanje.

1 Introduction

Power systems have become more complicated and open in response to the ever-changing power sector, which is defined by an increasing dependence on renewable energy sources with the integration of various grid-connected entities [1]. Power system operators face significant hurdles due to this development since they are required to make complex scheduling choices within an ever-expanding operating scope. Therefore, a game-changing answer has arisen in the shape of LLMs [2]—powerful deep learning frameworks educated on massive text datasets. These models are great at interpreting and making expressions that seem human, giving operators the skills, they need to handle complicated situations with ease. The incorporation of LLMs represents a notable stride forward in handling the intricacies and decision-making obstacles of contemporary power systems. Power systems in a

complicated data environment may be effectively navigated with the help of LLMs, which has skills in natural language processing, picture recognition, along with time series analysis [3]. They fortify the basis of scheduling as well as decision-making optimization by improving the extraction of crucial information from massive datasets. With their strong analytical and reasoning skills, LLMs can intelligently retrieve data and answer questions. This allows them to analyze a variety of inputs, including past load statistics, weather predictions, and real-time news. This makes a big difference in coming up with complex optimization plans. In addition, LLMs improve power system HCIs [4] by facilitating operators' ability to make educated, high-quality judgments via the comprehensible display of complicated data and operational conditions. The use of LLMs in power systems has many benefits, including improved dispatch accuracy and efficiency, more system flexibility and stability, new research opportunities, and exciting commercial potential. While there is a rising interest from influential entities in creating customized LLMs for electrical systems, there are also considerable security problems with this application [5]. The use of LLMs in power systems is becoming more open, which has many advantages but also has certain security risks, especially with regard to data security and the reliability of decisions [6], [7], [8]. Research and inquiry of these concerns are currently lacking. In order to fill this knowledge vacuum and guide the development of safer LLMs for use in power systems [9], [10], [11], this study conducts a thorough examination of the risks associated with LLM applications in these systems.

2 Security aspects of LLMs

Security flaws in LLM that allow rapid injection attacks are among the most critical ones. These types of attacks take advantage of LLMs' fundamental features by deliberately feeding them data that will cause them to operate in an unexpected way or leak private information [12], [13], which is depicted in Fig 1. Industries that deal with sensitive data are especially worried about the consequences of these vulnerabilities. In static analysis while (ii) dynamic analysis are the two conventional ways to find security flaws. Static analysis involves looking for bugs in the code without actually running it. Therefore, analysis does not account for the possible [14], [15] effects of the executable setting, which includes the OS and hardware. However, dynamic analysis can only reason regarding the observed execution routes and not all conceivable program paths; it executes the code to verify how the software would function in a run-time context. Therefore, both static and dynamic code evaluations each have their own set of issues. While still satisfying operational requirements, this method helps reduce the dangers associated with too strong models [16].

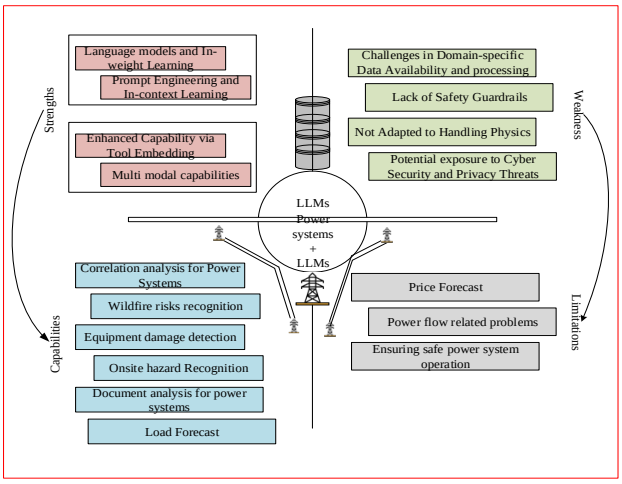

Figure 1: LLM for security threats

- **Infrastructure security:** Putting money into safe infrastructure is crucial. Place LLMs in secure, secluded areas and set up rigorous permissions for access. To greatly lessen the likelihood of illegal access or data interception, make sure that all communications with the LLM take place via encrypted communication channels.
- **Input sanitization and validation:** Any LLM security plan must include input validation and sanitization. Make sure that any harmful suggestions don't reach the LLM by implementing strong input filtering measures. Also, think about using anomaly detection systems driven by AI to spot suspicious input trends that might indicate an assault is underway.
- **Output filtering and monitoring:** Including input validation while sanitization in any LLM security strategy is essential. Use stringent input filtering procedures to prevent any detrimental recommendations from reaching the LLM. Anomaly detection systems powered by AI might also be useful for seeing strange input patterns that could signal an attack is happening.

A. Large language models in power systems

The merging of physical power systems with cutting-edge information and communication technologies allows for intelligent scheduling, real-time sensing, and quick responses as conventional power systems transform into cyber-physical power systems (CPPS) [6]. In this light, LLMs' incorporation into CPPS stands out as a major step forward in improving its intelligence capabilities and maximizing the efficiency of power system operations. As shown in Figure 1, LLM may be an important tool for assessing data from the physical system and assisting with cyber system decision-making. It is necessary to create customized LLMs that combine state-of-the-art AI training approaches with substantial power system domain knowledge before applying LLMs to power systems.

First, data is gathered and prepared according to industry standards. Then, a strong foundation model is adjusted so it can comprehend and provide power system-friendly language. Competence in machine learning as well as NLP is required, but so is an in-depth familiarity with the specific difficulties and subtleties of the power system. After the training phase is over, the next step is to validate and test the model thoroughly to make sure it works in real-world situations. The special benefits that LLMs provide in areas like as intelligent decision support in power systems, multimodal data analysis, and natural language processing originate from their in-depth comprehension and construction of human language. When it comes to the CPPS design, LLMs are most useful at the information and application levels of the cyber system. Substantial data is processed and analyzed at the information layer by use of sensors, smart meters, and various other such devices.

To predict future power needs, LLMs may look at things like weather reports, user habits, and past and present load data. At the application layer, LLMs let power system operators make better decisions, optimize power system operations, create scheduling strategies, and get useful information. In addition, LLMs may improve human-machine interaction using natural language processing technology, which adds a lot of intelligence to CPPS as a whole. Despite these advantages, the security of LLM incorporation into CPPS might be jeopardized by the dynamic complexity and openness of current power systems. The usual interpretability, openness, and specificity of optimization-based approaches to decision-making in power systems are usually lacking in this case. Security concerns around LLMs include the possibility of their abuse in cyberattacks, their vulnerability to data manipulation, and the effects on system stability. However, LLM decision-making is not always open and honest, which might jeopardize power system security and stability at crucial times. For this reason, it is critical for the secure functioning of CPPS to do further research into these possible dangers.

B. Security threats in power systems using LLMs

Operational integrity, Data privacy, as well as system vulnerabilities are just a few of the security risks that could arise from integrating LLMs into power systems. The following are a few ways in which these dangers might manifest.

Critical decision-support and information analysis services in power systems could be unavailable when required if this happens. The steady functioning and safety of the power system might be jeopardized, for example, if operators rely on LLMs for quick reaction or decision analysis during crises; a denial-of-service attack could lead to delayed or incorrect choices. And denial-of-service assaults may be a distraction or a camouflage for more serious attack tactics. Consequently, protecting electrical systems from DoS assaults necessitates improving LLM security. Strategies for managing traffic and keeping tabs on it, as well as creating LLMs that can withstand assaults like this, may be part of the solution.

There can be privacy and security concerns with using LLM in power systems. While LLM's sophisticated analysis and processing of data capabilities greatly enhance the power system's operating efficiency, they also present the possibility of privacy intrusions. This is mostly because LLMs are deployed as readily available resources inside the power system and are meant to increase cooperation efficiency across many departments. Because of their high level of accessibility, LLMs might be targeted by malicious actors. Attackers may get critical information regarding the power system, including operational data, control tactics, and security measures, if they gain access to LLMs via their clever question-answering system. In addition to breaching data security, this form of privacy theft may pave the way for more sophisticated assaults like fake data injection attacks (FDIAs). Nevertheless, FDIA is predicated on the idea that attackers are somewhat aware of the power system's operating circumstances in real-time in order to plan successful attacks. For attackers, acquiring such knowledge has always been a huge hurdle, making the goal hard to achieve in practice. But this obstacle may be much reduced with LLMs in the power systems of the future. The integrity of the power system may be jeopardized if attackers used LLMs to gather extensive operational information, which they could then use to develop FDIA methods. Thus, whereas LLMs improve the intelligence and efficiency of power systems, they also pose a danger of privacy breaches that might be taken advantage of in advanced assaults such as FDIA. Possible countermeasures include using data sanitization methods and restricting LLM access to important operational data. In order to reduce the security risk of information security breaches, it is necessary to filter and change the operational data so that it is still relevant for lawful purposes but cannot be used to create FDIA plans.

To guarantee performance, LLMs need a considerable investment in computer resources and training time since they are ultra-large-scale neural networks. It is critical to maintain their performance once installed in power systems. On the other hand, the operational integrity might be compromised if, in the long run, the power system makes decisions that are unsuitable or wrong due to changes in the LLM's internal parameters. There are two primary sources that might lead to performance deterioration that is important to operational integrity.

To begin, LLMs run the risk of learning false or misleading information if the data set (which incorporates training, validation, and evaluation sets) is intentionally

changed when training or fine-tuning. Such mistakes may affect the precision and dependability of decisions by causing the final model variables to deviate from the expected values. Second, the internal parameters of LLM are vulnerable to direct manipulation. The LLM's output might drastically differ from expectations if attackers get access to and change these settings after deployment. This would reduce the efficacy of decision-making and could cause major operational concerns. In either case, the stability and efficiency of the power system might be jeopardized because to incorrect judgments in system operations caused by worsening LLM performance. This highlights the critical importance of protecting LLM information and model parameters against deterioration in performance. To maintain the trustworthiness and integrity of LLM, stringent security measures are required during the whole training, deployment, and operating phases.

By coordinating activities across different departments and operators, LLM implementation may improve the power system's overall efficiency. But this also implies that the LLM can talk to a lot of terminals, which means that there will be a lot of open interfaces and human-machine interactions. Certain interfaces might be vulnerable to attackers in such a public setting. Attackers may execute SDAs against exposed interfaces in LLMs, posing a security danger to systemic vulnerabilities, since these systems may communicate with many operators via intelligent question answering. There are two methods for doing SDAs. To start with, you may trick the LLM into giving you inaccurate or irrelevant results by manipulating the data it uses for input (the query semantics). To get results for "historical load" instead of "real-time load," for instance, one may alter the query. Altering LLM's outputs directly to generate diverging answer semantics is the second approach. No matter the approach, SDAs pose a threat to operators' ability to make informed decisions on power system operations by providing them with inaccurate or misleading data. The electricity system's dependability and efficiency might be severely compromised by this false information. Consequently, safeguarding LLM inputs and results and keeping an eye on them is crucial for avoiding SDAs. To prevent attackers from using LLMs as a weapon to target the power system, it is crucial to establish stringent data validation as well as security rules within the system. This will guarantee that information is accurate and consistent.

However, LLMs bring additional security risks like as data privacy breaches and vulnerability to cyber threats such SDAs and DoS attacks, even if they are anticipated to greatly improve future power systems' operational efficiency along with decision-making capabilities. We need an all-encompassing, multi-dimensional framework to solve these security problems. Building strong LLM frameworks, including advanced anomaly detection procedures, and creating LLM structures with built-in security features are essential to this. All the way through an LLM's lifespan, it is critical to comply with data protection laws and ever-changing cybersecurity regulations. The implementation of adaptable security rules and regulations, together with human-in-the-loop tactics to strengthen LLMs against new security threats, is essential for reducing the impact of these developing security threats. In order to support these tactics, it is crucial to work with experts from other fields and validate them via real-world testing. These efforts are essential for electricity systems to adapt to LLMs, integrate with their improved capabilities, and meet reliability and security criteria. Improving cybersecurity, establishing data protection rules, and developing ethical use guidelines for LLMs should be top priorities for anyone involved in the electricity industry. It is necessary to teach employees thoroughly and work together with other businesses and government agencies to promote a culture of security awareness while readiness. To combat any security risks linked to LLMs, it is crucial to do regular risk assessments, meticulously monitor, and upgrade systems periodically. Using LLMs to future electrical infrastructure requires this comprehensive framework, which seeks to balance the promise of LLMs alongside the mitigation of security vulnerabilities. To maintain this balance, we need to do proactive implementations of research, and design LLM systems that are safe and transparent while still meeting regulatory criteria.

Understanding the present applications, problems, and possibilities of LLMs is crucial for providing a thorough overview of their usage in software vulnerability and cyber security research. This is why we've set out to compile a comprehensive literature study on LLMs' use in this sector. Therefore, the following research issues are intended to be addressed by this study:

- **RQ1**: For software vulnerability and cyber threat detection, why should we employ LLMs?
- **RQ2:** How exactly do software vulnerabilities while cyber security risks detection make use of LLMs?
- **RQ3:** What role may LLMs play in identifying and mitigating cyber security risks and software vulnerabilities?
- **RQ4:** In the context of finding and dealing with cyber security risks and software vulnerabilities, how does an LLM model work?
- **RQ5:** When training LLMs to identify software vulnerabilities, what kinds of data sets work best?
- **RQ6:** In terms of finding and dealing with software vulnerabilities along with cyber security concerns, how effective are LLMs compared to more conventional approaches and tools?
- **RQ7:** In order to evaluate LLMs, what measures are taken to deal with cyber risks and software vulnerabilities?

- **RQ8:** When it comes to cybersecurity, what are the obstacles to using LLMs? Is there a way to make LLM better at finding software vulnerabilities and cyber threats?

3 Related work

By tightly integrating LLM-based reasoning with robotic autonomy, the authors of [16] demonstrated the first airborne intelligent agent that could execute tasks in an open environment. Two primary constraints are addressed by our co-designed hardware-software system: (1) A 14B-parameter model can be inferred at 5-6 tokens/sec using an edge-optimized computing platform; (2) A bidirectional cognitive architecture can be implemented to combine slow deliberative planning (LLM task planning) using fast reactive control (state estimation, obstacle avoidance, mapping, while motion planning). The system's ability to reliably plan tasks and interpret scenes in settings with limited communication is supported by early findings obtained with our prototype. These environments include sugarcane tracking, power grid inspection, mining tunnel exploration, as well as biological observation applications. By combining elements of job planning with those of robotic autonomy in open environments, this study creates a new paradigm for embodied airborne AI.

In order to acquire a QA dataset containing fundamental information about power transformers, the researchers in the article [17] first suggest an automated technique for dataset building based on LLMs. After then, the LLM undergoes low-rank adaptive fine-tuning using this dataset. Afterwards, a database of external knowledge is constructed, which includes both basic information and examples of power transformer failure. The knowledge replies were created by RAG and the fine-tuned LLM using prompt texts. This work proposes a strategy that, according to experimental data, promotes the implementation of LLMs in the power sector by producing more succinct and expert replies than QA systems powered by conventional LLMs.

Among the many operational responsibilities related to power systems, the authors of [18] presented Grid Artificial Intelligent Assistant, a groundbreaking Large Language Model developed to aid in adjustment of operations, monitoring of operations, and handling of black start situations. For GAIA to function at its best in this area, we came up with a new way to build datasets that uses a variety of data sources. The use of multidimensional data in power system administration may be made smooth using this method, which simplifies LLM training. To further improve GAIA's input-output efficiency in dispatch circumstances, we have developed customized quick techniques. When tested on the ElecBench benchmark, GAIA outperforms the baseline model, LLaMA2, on many criteria. In real applications, GAIA has proved its capacity to increase decision-making processes, enhance operational efficiency, and promote enhanced human-machine interactions in power dispatch processes. Future developments in this sector will be facilitated by this article, which proves the practical value of LLMs and widens their application to power dispatch.

By using LLMs to predict missing measurements as well as offering pseudo-measurements, the authors of [19] tackle these difficult problems and present an edge learning framework for DSSE that uses forecast-then-estimate. In order to identify patterns from previous data and provide reliable predicting results, the suggested LLM integrates measurement sequences with natural language-based cues. To further strengthen state estimation in the face of missing measurements, a neural network model based on convolutional layers is secondly presented. The third step is to restructure the deep learning-based DSSE into a multi-task learning structure with shared as well as task-specific layers. This will help with overfitting. When trying to strike a compromise between competing priorities, the uncertainty weighting technique comes in handy. To prove that the suggested forecast-then-estimate architecture works, the Simbench example is used in numerical simulations.

In order to validate program behaviors throughout aspect weaving, the experimenters in [20] suggested an AI-enhanced adaptive monitoring framework that combines AOPIs using LLMs, specifically GPT-Codex AI, to create and optimize statistical models and monitoring aspects in real-time. As a result, NL interaction, adaptive model verification, and smart run-time analysis are all made possible. We built a dataset for analysis and tested the system on 10 different Java classes from JHotdraw 7.6 through extracting numerical and context data. The findings demonstrated that the framework preserved the Java OOP class's integrity while offering predictive insights into possible conflicts and optimizations via the dynamic refinement of features and models using observed behavior. With a 37% decrease in false positives and a 94% accuracy in recognizing possible conflicts, the results show that the framework is effective in detecting aspect weaving's subtle behavioral changes. In addition, developers may get practical, understandable reasons for flagged behaviors using NL interfaces with explainable AI integration, which boosts interpretability and confidence in the system.

Using a simulated dataset captured during the monitoring of an electric vehicle's battery management system, the researchers of [21] evaluate the performance of eight open-source LLMs in detecting errors in cyber-physical systems. Our work seeks to investigate the possibility of using open LLMs for defect identification by using pretrained LLMs without fine-tuning by combining retrieval augmented generation approaches with textual

encoding methods. In terms of accuracy, recall, and Fl-score measures, our findings demonstrate that open LLMs are capable of defect detection, with Mistral surpassing competing models like Mixtral, codellama, and Gemma. In addition, our findings emphasize the significance of textual encoding techniques for improving LLMs' fault detection skills; these algorithms have some explanatory power about the abnormalities that are found. This study paves the way for further investigation into improving fault detection and localization in cyber-physical systems by showing that open LLMs may be used for this purpose.

4 Methodology

Taking the source code as input, our ML model determines whether it includes particular preset vulnerabilities or not. This allows us to treat the vulnerability prediction issue as a binary classification job, which is given in fig 2.

4.1 Dataset description

The HAI dataset has been revised several times. The samples were collected over the course of 19 days from a testbed that was supplemented with an HIL simulator that mimics steam-turbine power generation while pumped-storage hydropower generation. The first five days were devoted to collecting sensor data during normal operations, and the remaining fourteen days were devoted to collecting sensor data during attacks. In this case, we specifically use HAI 21.03, that was released in 2021. By tampering with packets, an attacker in the HAI dataset consistently introduces physical measure inaccuracies. A total of twenty-five distinct forms of assault. The HAI dataset was acquired using a realistic ICS testbed that was supplemented by a Hardware-In-the-Loop (HIL) simulator. This simulator duplicates the power generating processes of steam-turbine and pumped-storage hydropower. A FESTO water treatment system, an Emerson boiler, and a GE turbine are all part of the HAI testbed's physical control systems, which are linked by a dSPACE hardware-in-the-loop software. We developed a system that can remotely and autonomously regulate a feedback control loop. We obtained HAI dataset 1.0 in this setting by performing many benign and malicious situations over an extended length of time with little human intervention. HAIEnd is a dataset that gathers tag values for the boiler DCS's internal logic. It is gathered concurrently using the same version of the HAI dataset during the same experiment, which is given in table 1.

Table 1: Statistics of evaluation datasets

Datasets	**Features**	**Training Samples**	**Testing Samples**	**Prop. of Anomalies**
HAI	79	921,603	402,005	2.2%

A number of iterations of the HAI dataset exist. In this case, we employ HAI 21.03, that was released in 2021. The samples were gathered over the course of 19 days from a testbed that was supplemented with an HIL simulator that mimics the generation of power by steam turbines and hydropower by pumping storage. During the first five days, sensor data was collected under normal operating conditions, and during the remaining fourteen days, data was collected under attack conditions. An adversary routinely manipulates packets in the HAI dataset, leading to physical measure inaccuracies. We have a total of 25 different sorts of attacks, which is given in Table 2.

Table 2: HAI feature set

HAI	P2_VTR02, P1_FCV02D, P3_LL, P2_AutoGo, P1_PP01BD, P3_LH
	P1_PP01AD, P1_PP01AR, P2_OnOff, P2_RTR, P2_ManualGo
	P1_PCV02D, P2_VTR01, P2_VTR04, P4_HT_PS, P1_PP02D
	P4_ST_PS, P1_PP02R, P2_MSD, P2_VTR03, P2_TripEx, P1_STSP

Data fields

Each CSV file contains time-series data that is consistent with one another. The time in the format "yyyy-MM-dd hh:mm:ss" is shown in the first column, while the other columns include the recorded SCADA points of information. Data labels for the occurrence of attacks are provided in the final four columns. The attack column applies to every process, but the remaining three columns are exclusive to the control processes that correspond to them, which is given in table 3.

Table 3: CSV Data

time	**P1_B2004**	**P2_B2016**	**...**	**P4_HT_LD**	**attack**	**attack_P1**	**...**	**attack_P3**
20190926 13:00:00	0.09830	1.07370	...	0	0	0	...	0
20190926 13:00:01	0.09830	1.07410	...	0	1	0	...	1
20190926	0.09830	1.07380	...	0	1	0	...	1

13:0 0:02								
201 909 26 13:0 0:03	0.09 830	1.07 360	. . .	0	1	1	. . .	1
201 909 26 13:0 0:04	0.09 830	1.07 430	. . .	0	1	1	. . .	1

4.2 Data preprocessing

For efficient processing of industrial power data, we recommend two preprocessing steps: data normalization with constant feature elimination.

- Data normalization (Normalization): One of the preprocessing methods, it involves transforming or scaling the values of every attribute so they contribute equally. Data normalization is necessary in an industrial power dataset due to the fact that many characteristics have distinct scales.
- Constant feature exclusion (Feature exclusion): Since they aren't required to construct a machine learning model, constant features (also known as unchanging features) aren't considered in our work. Hence, avoiding training with certain characteristics is one option.

4.3 Feature scaling

Both the size and complexity of our CVE dataset grew substantially after encoding. Changes to the relative sizes of our features were introduced at this stage. The models might have been skewed since features like "Age," "Total Salary if Available," and "Exact CGPA" had various ranges and units. Consequently, we scaled the features such that they all had an equal impact on the results. Data range and distribution are adjusted during scaling to ensure that all features are on the same scale. As a principal strategy, we settled on normalization scaling.

Data normalization reduces the range of possible values to [0, 1], which is useful for algorithms like gradient descent-based approaches that are sensitive to the magnitude of feature values. To normalize continuous variables such as "Weekly Study Hours," "Reading Frequency (Scientific Books/Journals)," and "Reading Frequency (non-Scientific Books/Journals)," we used min-max. We made sure that each feature contributed proportionally to the study by rescaling it, so that no one feature could dominate because of its size. Data normalization reduces the range of possible values to [0, 1], which is useful for algorithms like gradient descent-based approaches that are sensitive to the magnitude of feature values. To normalize continuous variables such as "Weekly Study Hours," "Reading Frequency (Scientific Books/Journals)," and "Reading Frequency (non-Scientific Books/Journals)," we used min-max. We made sure that each feature contributed proportionally to the study by rescaling it, so that no one feature could dominate because of its size.

$$\text{Xnormalized} = \frac{\text{X}-\text{Min(X)}}{\text{Max(X)}-\text{Min(X)}} \qquad (1)$$

In cases where the following words are:

- The initial value of the variable is denoted by X.
- Xnormalized is the variable's normalized value.
- In each occurrence of the variable X, the minimum value is represented by Min(X).
- Max(X) is the largest value of X that has ever been, in every case.

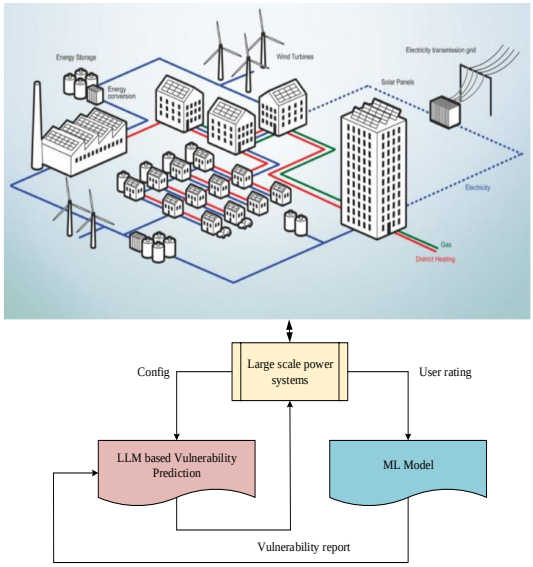

Figure 2. LLM with Security Model

4.4 Selection of LLMs for the study

Considering the variety of language models available, it is crucial to establish explicit criteria for choosing the ones that align most closely with the study aims. The following factors were taken into account while selecting big language models to be tested in this study:

- Figure out the model's size and scale. An important factor in the model's capacity to produce and comprehend text is its size, and more specifically the quantity of parameters. A great deal of complexity along with contextual significance may be generated by large models having billions of parameters. Nevertheless, it is important to take into account the substantial

computing resources required by these models when choosing them for this study.

- Works well for designated purposes. Consideration of the model's applicability to individual jobs should guide the selection process. A critical need in this scenario is the model's capacity to produce massive volumes of text.
- Access and licencing. The models need to be freely accessible for researchers to use.

Mathematical background

Mathematically, LLMs are based on neural networks, and more specifically, transformer structures. These models may determine the relative relevance of words in a phrase or document by processing text sequences using self-attention processes. Mathematical foundations include word vector representations (embeddings), sequence order maintenance via positional encoding, and multi-head attention enabling model targeted attention in input sequence prediction. During training, a loss function—usually cross-entropy for language tasks—is minimized by altering millions—or perhaps billions—of parameters via gradient descent.

GPT-3

In the GPT-3 model, the self-attention process is mathematically shown as:

$$\text{Attention}\ (\mathrm{Q},\mathrm{K},\mathrm{V}) = \text{softmax}\left(\frac{\mathrm{QK}^{\mathrm{T}}}{\sqrt{\mathrm{d}_{\varepsilon}}}\right)\mathrm{V}$$

where d_{k} is the dimensionality of the keys and Q, K, and V are the queries, keys, and values matrices that are obtained from the input embeddings, respectively. The softmax algorithm guarantees that all input attention weights add up to 1.

BERT

Instead of processing words independently, BERT analyses their context inside a phrase, making use of transformer architecture in a novel way. The bidirectional attention mechanism in BERT is responsible for this. From a mathematical perspective, this is accomplished by assigning attention scores to each word, utilizing a softmax function to prioritize these scores, and finally, utilizing these values to generate contextually enhanced word embeddings. The softmax of the dot result of the query with key vectors, divided with a scaling factor, is the central formula that affects the embeddings of the final output.

T5

The T5 model, which stands for Text-to-Text Transfer Transformer, makes use of the transformer's encoder-decoder design. Its unique selling point is that it simplifies inputs and outputs for all natural language processing tasks by seeing them as text-to-text problems. Akin to other transformers, the key mathematical procedures include self-attention mechanisms that compute attention scores to ascertain the interrelationships between various textual elements. Additionally, it makes advantage of the decoder's cross-attention methods to zero in on pertinent portions of the encoder's output. The model's training on a "span corruption" objective—predicting input text gaps—allows it to generalize well across other natural language processing tasks.

The latest version of TinyLlama Chat, an AI model that minimizes resource use without sacrificing speed, is 1.1. It is a more compact variant of models that use the LLaMA architecture for processing natural language. With TinyLlama, we want to make big models powerful with a lot less parameters, so we can save computing resources without sacrificing speed. For this reason, it was chosen primarily for this investigation.

4.5 Security analysis using ML

The creation of data is both connected to time series, as we have seen. To assess the three models' accuracy, this research used LSTM and GRU neural network models for one-day-ahead ower forecasting, along with the classic LLM model.

4.5.1 Artificial Neural Network (ANN)

When it comes to deep learning, an artificial neural network is a key tool. The three layers that make it up are the input, hidden, and output layers. Artificial input neurons make up the input layer, which is responsible for bringing raw data into the system and sending it on to higher-level artificial neural processing layers. An artificial neuronal network receives a series of weighted inputs along with generates an output via an activation function in the hidden layer, which is located between the input and output layers. As the last neuronal layer, the output layer is responsible for generating the program's outputs. To increase the model's performance, the activation function assesses the relevance of the neuron output for each layer of neurons. A benefit of artificial neural networks (ANN) is their exceptional adaptive ability and its capacity to handle complex nonlinear challenges. Consequently, ANN is a widely used approach to making forecasts a reality. When there is a dearth of data, however, performance suffers.

4.5.2. Long Short-Term Memory (LSTM)

One kind of neural network that struggles to process data over extended periods of time is the RNN, or Recurrent Neural Network. This kind of network is prone to the disappearing and expanding gradient problems. Therefore, Hochreiter invented long short-term memory and suggested that LSTM can train-side handle the disappearing and exploding gradient issue. The forget gate, which determines the percentage of information that requires to be maintained in the memory cell, was included into the LSTM architecture. The more data there is, the better LSTM performs.

4.5.3. Gated Recurrent Unit (GRU)

In this work, we first proposed the gated recurrent unit. Although GRU lacks an output gate, its architecture is otherwise comparable to that of LSTM. It employs an update gate with a reset gate in an attempt to resolve the vanishing gradient issue. Due to its quick calculation speed and limited number of gates, GRU is preferable than LSTM when working with small datasets.

The most popular statistical tool for determining whether or not there is a linear connection between manual and projected grades is Pearson's correlation. Divide the product of the two variables' standard deviations by their covariance to get Pearson's correlation coefficient.

To find the optimal values for the hyper-parameters used in each model's creation, hyper-parameter tuning was performed. In order to find the optimal values for a model's hyperparameters, we utilized the Grid-search method, which involves searching exhaustively over a set of values for each estimator. The training set's weight vectors are updated at regular intervals called epochs. In the model's optimization experiment, we sought to determine the ideal epoch, which might be anywhere from 10 to 40. How many samples are used to modify the model weight in the training set is called the batch size. From twenty to two hundred and fifty, we looked for the sweet spot for batch size. When making a prediction based on feature values, the window size determines how long the input time steps are. From 5 to 1000, we looked for the sweet spot for window size. The learning rate is the increment in step size used to decrease the gradient of errors. We experimented with learning rates as low as 0.0001 and as high as 0.05 in an effort to identify the sweet spot. RNNs include LSTMs and GRUs among their varieties. Their "memory" of previous occurrences allows them to handle data sequentially. For power monitoring structures, this is of the utmost importance, since the identification of risks relies heavily on the context along with temporal relationships of data points, such as a sequence of orders or changes in sensor readings over time.

1. **Modeling Normal Behavior**: While running in a safe environment, the model is trained only on data collected by the power monitoring device. Here, the LSTM or GRU figures out the usual sequences, patterns, and statistical features of the data. One example is that it figures out the typical order of operations for a remote terminal unit (RTU) to communicate with a SCADA system.
2. **Anomaly Detection**: In order to anticipate the next condition or data point in a sequence, the trained model is used during real-time monitoring. An anomaly is shown when the actual information differs substantially from the model's forecast. This is predicated on the idea that a fresh vulnerability or attack would cause a chain reaction that the algorithm has never seen before, rendering it unable to precisely predict.
3. **Vulnerability Inference**: Finding a vulnerability is not the only possible outcome of detecting an abnormality. After then, the anomaly is examined to find out what may have caused it. At this point, an LLM may be included to examine the marked data, connect it to other system details, and provide a straightforward description of the possible vulnerability, such as attempt to manipulate data or insert commands.

Advantages of LSTMs and GRUs

- **Handling Sequential Data**: Understanding the context of a cyber-attack requires LSTMs and GRUs to capture both short- and long-term dependencies in time-series data. These models can detect whether a string of apparently innocuous events is really part of a more sinister pattern.
- **Reduced False Positives**: When compared to more basic rule-based systems, these models are able to drastically cut down on false positives by learning the system's typical behavior. Their understanding of "normal" has evolved to include more nuances, so they are less prone to mistakenly labeling a seemingly innocuous occurrence as dangerous.
- **Computational Efficiency**: In example, with fewer parameters, GRUs are computationally easier to implement than LSTMs. Their ability to respond quickly makes them ideal for use in systems that track data in real-time. To combine the advantages of both long-term memory LSTMs and GRU efficiency, several researchers have suggested hybrid LSTM-GRU models.

5 Experiments

A 64-bit Windows 10 PC containing an Intel(R) Core (TM) i7-8650U CPU @ 1.90GHz and 32.0 GB RAM is

used for all the studies related to our suggested technique. Given that the code2vec developers made their instructions compatible with Ubuntu systems, we ran our tests on a virtual machine running Ubuntu 18.04 with 24022 MB of base memory and 4 CPUs. This machine is identical to the Windows PC described here.

Finding the sweet spot for a hybrid LSTM and GRU model's hyperparameters is all about doing a thorough search for the greatest possible combination of parameters on each particular dataset. A deliberate strategy is required to handle the enormous number of alternative configurations since these designs are sequential and frequently complicated, which is given in Table 4.

Table. 4 Hyperparameters LSTM-GRU

LSTM-GRU	Batch Size	517
	Learning Rate	0.003
	Activation Function	tanh
	Output-Units	2
	Output-Type	Single Label
	No of epochs	15
	No of Hidden Layers	06
	Output-Layer-Activation-Function	linear
	Optimizer	Adam
	Loss Function	mean squared error
	Hidden Units	275
	Dropout Ratio	0.3

5.1 Performance analysis using machine learning

- Gradient Boosting: As an ensemble method in machine learning, Gradient Boosting repeatedly combines the results of several weak models to create a robust prediction model. Its primary goal is to reduce mistake rates via redistributing the weights of erroneously categorized cases. By focusing on the areas where the present model is lacking, this strategy improves predictions over time, resulting in a final model that is both resilient and accurate.
- K-Nearest Neighbors (KNNs): Knowledge-based nearest neighbours (KNNs) is an easy-to-understand approach for data point classification. An unknown data point is labelled according to the labels of its nearby neighbors in the training dataset. Non-linear as well as locally clustered data are where KNNs really shine.
- Naive Bayes: A probabilistic technique for classification, Naive Bayes is based on Bayes' theorem. Although it may not always be applicable to real-world data due to its feature independence assumption, it nonetheless manages to perform very well in a number of applications. Text categorization and spam filtering are two common applications of Naive Bayes because of its computational efficiency.
- Random Forest: As part of its ensemble-learning approach, Random Forest builds several decision trees throughout training and then averages their predictions to provide an output. Every tree learns to generate predictions on its own using a different subset of the training data. A dependable and flexible approach for regression and classification applications, Random Forest averages or votes on the outputs of each individual trees.

5.2 Evaluation measures

We used four different metrics—accuracy, precision, recall, and F1-score—to measure the models' performance in our experiments. Among these metrics, accuracy has been used in a plethora of previous studies. The F1-score provides a thorough evaluation of the classifier's performance in each class separately. It combines accuracy and recall, making it useful in situations with different class distributions. We classified software occurrences into four groups: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) throughout our investigation. In summary, our research made use of the following assessment measures:

- Accuracy—this refers to the classifiers' ability to correctly evaluate the students' performance based on the training dataset. It is determined using the formula in (2).

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+FP+FN+TN}} \quad (2)$$

- The accuracy rate of positive predictions is known as precision-it. As seen in (3), it is computed.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (3)$$

- Recall—also known as the true positive rate (TPR), it is the proportion of data samples that an ML model correctly identifies as containing each IDS event. The formula for this is given in (4).

$$\text{Reaill} = \frac{\text{TP}}{\text{TP+FN}} \quad (4)$$

- The most important statistic for cloud intrusion detection is F1-score-it, a composite measure that combines accuracy and recall. Equation (5) shows the calculation.

$$\text{F1Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (5)$$

- One of the most popular ways to quantify the accuracy of forecasts is by looking at the root mean square error or root mean square deviation.

The deviation of the predicted values from the actual values, as assessed using the distance formula, is shown. As a great general-purpose error measure for numerical forecasts, RMSE sees extensive application. Here is how RMSE is computed:

$$\text{RMSE} = \sqrt{\text{MSE}},$$

On the other hand, mean square error (MSE) calculates the average of the squares of the discrepancies between the predicted and actual values. The formula for MSE is this:

$$\text{MSE} = \frac{1}{\text{N}}\sum_{\text{i=1}}^{\text{N}} (y_1 - \hat{y}_1)^2$$

Table 5: Model performance for original dataset

Model Name	**LLMs**	**Split Size**	**RMSE**	**F1-Score**	**Accuracy**	**Pearson Correlation**
SVM	BERT	70–30	1.61902	0.19927	0.22287	0.21246
Gradient Boosting			1.04117	0.65803	0.66862	0.63649
Naive Bayes			1.06047	0.66809	0.64516	0.66427
Random Forest			0.99158	0.60616	0.64367	0.64367
KNN			0.89391	0.70164	0.70614	0.45196
DT			1.08879	0.63096	0.64474	0.47758
LR			1.31466	0.58836	0.58333	0.43599
Proposed			0.99299	0.63995	0.65351	0.43184
SVM	GPT 3	80–20	1.91571	0.42417	0.40351	0.25566
Gradient Boosting			1.31803	0.67083	0.67105	0.59558
Naive Bayes			1.24106	0.73221	0.72807	0.65142
Random Forest			1.19878	0.58192	0.56579	0.64576
KNN			1.22907	0.68833	0.68859	0.76269
DT			1.4047	0.67267	0.67105	0.71517
LR			1.79938	0.50837	0.49123	0.62883
Proposed			1.15167	0.63541	0.64474	0.71271

Table 5 shows the performance analysis of the proposed methods. In all instances, the F-score was determined to be over 70%, suggesting that using software metrics might be a viable approach for vulnerability prediction. Based on our findings, the Random Forest using 100 trees seems to be the optimal strategy, since it not only shows high Precision (over 90%) but also pretty good F-scores. This proves that the model adequately handles the issue of many false positives, a recognized issue in the research that limits the usefulness of the generated models. To be more precise, when the accuracy is poor, it implies the model generates a lot of False Positives. This forces the developer to pay attention to parts (like functions) that the model flags as susceptible even when they aren't really vulnerable. On top of that, finding a susceptible function would require the developer to assess a huge number of clean functions. Clearly, this results in a loss of important resources, namely the time and energy needed to identify a security hole. To further evaluate their efficacy, we ran supplementary trials with bigger models. This summary also shows that balancing the dataset via data augmentation helps ML models perform better. With fewer parameters and less space to work with, processing times are reduced. Compared to tests with smaller models, the findings achieved for F1-score, accuracy, as well as Pearson correlation are superior. However, in regard to the average running time, there is no significant difference in performance, which is given in fig 3.

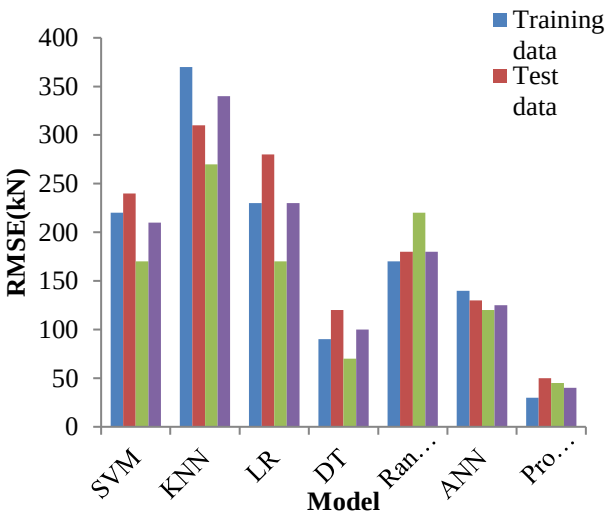

Figure: 3 RMSE analysis

There is a statistically significant difference between FNR and FPR values across all five models. Our best guess is that the algorithms are well-trained on typical data and can understand its features, but they aren't robust enough to identify novel, undiscovered assaults.

In a similar vein, when we compare the models' performance with normal and attack samples, our classification results reveal reduced TP and slightly greater TN. Due to our model's exclusive training on the normal dataset, we acknowledge that detecting assault samples would be more difficult. Consequently, each model does a better job of identifying normal samples than attack samples. On the Software Assurance Reference Dataset, BERT attains an F1-score of 0.9659 and an accuracy of 0.9507 while using recommended data processing and quick design in the interaction. These results show that vulnerability detection systems may benefit from reasoning-enabled Large Language Models, which can enhance both detection performance and interpretability.

When combined with LSTMs and GRUs, LLMs provide a huge improvement in the battle against power monitoring systems' undisclosed weaknesses. These systems may improve the dependability and resilience of vital energy infrastructure by merging their abilities in contextual comprehension, sequential data processing, with anomaly detection. The only way to fully protect our power networks from new cyber threats is to keep investing in R&D in these areas so that we can eventually overcome the obstacles we have already encountered. Our next step is to examine how the quantity of the dataset impacts the accuracy of the anomaly detection algorithms. We divided the training dataset into two halves based on the order of the observations, and our experimental design matches that. Here are some conclusions drawn from our experiments with different dataset sizes:

- We discovered that a portion of the whole training set—say, 40% or 60%—could provide results that were on par with the whole dataset. The pattern of periodic normal behaviors seen in the HAI dataset lead us to believe that this is the reason. Anomaly detection models may be built using a portion of the full training set, which allows for good detection accuracy.
- As the size of the training set grows, so does the time it takes to train all of the models. In cases where minimization of training time is paramount, the size of the training set may be used to exert control on the training duration of the model. To keep error rates low over time, model retraining might be required. Finding the sweet spot for training sample sizes can help keep retraining costs down.

6 Conclusion and recommendations

Among the most serious are LLM security holes that permit injection attacks with little delay. By intentionally feeding LLMs material that will cause them to perform in an unexpected manner or expose sensitive information, these sorts of attacks take use of LLMs' inherent properties. The potential fallout from these vulnerabilities is a major concern for industries that handle sensitive data. A thorough plan for LLM security must be developed in order to lessen these dangers. Creating trustworthy methods for model training and selection is the first stage. The advent of commercially available systems such as ChatGPT has led to the widespread adoption of several LLMs. Semantic search, which can do searches taking word meanings into account, has been captivated by this. In this case, we used TinyLlama Chat 1.1 to construct an LLM model. Using the extracted context and processed packet data as input, the LLM produces an easy-to-understand summary of the packet file. The application gives a clear, succinct, and well-organized summary of the network's actions using machine learning models.

References

[1] Barany, A., Nasiar, N., Porter, C., Zambrano, A.F., Andres, J.M., Bright, D., Shah, M., Liu, X., Gao, S., Zhang, J., Mehta, S., Choi, J., Giordano, C., & Baker, R.S. (2024). ChatGPT for Education Research: Exploring the Potential of Large Language Models for Qualitative Codebook Development. International Conference on Artificial Intelligence in Education. https://doi.org/10.1177/23328584251389621

[2] Stein, K., Mahyari, A.A., Francia, G.A., & El-Sheikh, E. (2024). Towards Novel Malicious Packet Recognition: A Few-Shot Learning Approach. MILCOM 2024 - 2024 IEEE Military Communications Conference (MILCOM), 847-852. DOI:10.1109/MILCOM61039.2024.10774059

[3] Yang, Z., Jin, Y., Liu, J., Xu, X., Zhang, Y., & Ji, S. (2025). Research on Cloud Platform Network Traffic Monitoring and Anomaly Detection System based on Large Language Models. DOI:10.1109/CISCE65916.2025.11065413

[4] Yan, Y., Hu, T., & Zhu, W. (2024). Leveraging Large Language Models for Enhancing Financial Compliance: A Focus on Anti-Money Laundering Applications. 2024 4th International Conference on Robotics, Automation and Artificial Intelligence (RAAI), 260-273. DOI:10.1109/RAAI64504.2024.10949516

[5] Gupta, S.K., Basu, A., Nievas, M., Thomas, J., Wolfrath, N., Ramamurthi, A., Taylor, B., Kothari, A.N., Schwind, R., Miller, T.M., Nadaf-Rahrov, S., Wang, Y., & Singh, H. (2024). PRISM: Patient

Records Interpretation for Semantic clinical trial Matching system using large language models. NPJ Digital Medicine, 7. 10.1109/RAAI64504.2024.10949516

[6] Dong, H., Zhu, J., & Chung, C. (2023). Prompts of Large Language Model for Commanding Power Grid Operation. Artificial Intelligence, Social Computing and Wearable Technologies. https://doi.org/10.1016/j.rcim.2026.103269

[7] Rao, M.R., & Sundar, S. (2024). Enhancement in Optimal Resource-Based Data Transmission Over LPWAN Using a Deep Adaptive Reinforcement Learning Model Aided by Novel Remora with Lotus Effect Optimization Algorithm. IEEE Access, 12, 76515-76531. DOI:10.1109/ACCESS.2024.3406749

[8] Bhaumik, R., Srivastava, V.K., Jalali, A., Ghosh, S., & Chandrasekharan, R. (2023). MindWatch: A Smart Cloud-based AI solution for Suicide Ideation Detection leveraging Large Language Models. medRxiv. DOI:10.1101/2023.09.25.23296062

[9] Araujo, A.S., Mercês, J.M., Silva, R.L., Alencar, A.V., Passos, I.F., Sousa, M.P., Dias, M.C., Meneses, T.F., & Santos, D.F. (2024). An Agentic Approach For Dynamic Software-Defined Network Management Using Large Language Models. 2024 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), 221-226. DOI:10.1109/NFV-SDN61811.2024.10807498

[10] Sanguinetti, M., Pani, A., Perniciano, A., Zedda, L., Loddo, A., & Atzori, M. (2024). Assessing Italian Large Language Models on Energy Feedback Generation: A Human Evaluation Study. CLICIT. https://doi.org/10.30574/gscarr.2024.21.2.0408

[11] Elsharef, I., Zeng, Z., & Gu, Z. (2024). Facilitating Threat Modeling by Leveraging Large Language Models. Proceedings 2024 Workshop on AI Systems with Confidential COmputing. DOI:10.1109/ICSES63445.2024.10763024

[12] Yao, Z. (2024). Formal Trust and Threat Modeling Using Large Language Models. 2024 Annual Computer Security Applications Conference Workshops (ACSAC Workshops), 232-239. DOI:10.1109/ACSACW65225.2024.00033

[13] Gupta, B.B., Gaurav, A., & Arya, V. (2024). Navigating the security landscape of large language models in enterprise information systems. Enterprise Information Systems, 18. DOI:10.1080/17517575.2024.2310846

[14] Tan, L., Xiang, Y., Tang, B., Li, H., Du, Z., Lu, Y., Ma, H., Xi, Z., Yang, J., Wang, S., & Li, L. (2024). Large Language Model based Framework for Secure Operation of Power Systems. 2024 3rd International Conference on Power Systems and Electrical Technology (PSET), 695-699. DOI: 10.70322/sesr.2025.10005

[15] Yang, Z., Jin, Y., Liu, J., Xu, X., Zhang, Y., & Ji, S. (2025). Research on Cloud Platform Network Traffic Monitoring and Anomaly Detection System based on Large Language Models. DOI:10.1109/CISCE65916.2025.11065413

[16] Xue, J., Deng, Y., Gao, Y., & Li, Y. (2024). Auffusion: Leveraging the Power of Diffusion and Large Language Models for Text-to-Audio Generation. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 32, 4700-4712. DOI:10.1109/TASLP.2024.3485485

[17] Fan, W., Zhao, Z., Li, J., Liu, Y., Mei, X., Wang, Y., Tang, J., & Li, Q. (2023). Recommender Systems in the Era of LLMs. IEEE Transactions on Knowledge and Data Engineering, 36, 6889-6907. https://doi.org/10.1109/TKDE.2024.3392335

[18] Otal, H.T., & Canbaz, M.A. (2024). LLM Honeypot: Leveraging Large Language Models as Advanced Interactive Honeypot Systems. 2024 IEEE Conference on Communications and Network Security (CNS), 1-6. DOI:10.1109/CNS62487.2024.10735607

[19] Kande, R., Pearce, H.A., Tan, B., Dolan-Gavitt, B., Thakur, S., Karri, R., University, J.R., Wales, U.O., Calgary, U.O., & University, N.Y. (2023). (Security) Assertions by Large Language Models. IEEE Transactions on Information Forensics and Security, 19, 4374-4389. DOI:10.1109/TIFS.2024.3372809

[20] Huang, C., Li, S., Liu, R., Wang, H., & Chen, Y. (2023). Large Foundation Models for Power Systems. 2024 IEEE Power & Energy Society General Meeting (PESGM), 1-5. DOI:10.1109/PESGM51994.2024.10688670

[21] Zhang, R., Du, H., Liu, Y., Niyato, D., Kang, J., Xiong, Z., Jamalipour, A., & In Kim, D. (2024). Generative AI Agents With Large Language Model for Satellite Networks via a Mixture of Experts Transmission. IEEE Journal on Selected Areas in Communications, 42, 3581-3596. DOI:10.1109/JSAC.2024.3459037

[22] Xu, H., Kang, Z., Zhang, Y., Jin, Z., Wang, M., Ge, L., & Lu, H. (2024). Research on the Construction of Knowledge QA System Driven by Large Language Model: One Case Study of Power Transformer Domain. *2024 7th International Conference on Data Science and Information Technology (DSIT)*, 1-8. DOI:10.1109/DSIT61374.2024.10881483

[23] Cheng, Y., Zhao, H., Zhou, X., Zhao, J., Cao, Y., Yang, C., & Cai, X. (2025). A large language model for advanced power dispatch. Scientific Reports, 15. DOI:10.1038/s41598-025-91940-x

[24] Xie, R., Yin, X., Li, C., Chen, G., Liu, N., Zhao, B., & Dong, Z.Y. (2024). Large Language Model-Aided Edge Learning in Distribution System State Estimation. IEEE Internet of Things Journal, 12, 25966-25976. DOI:10.1109/JIOT.2024.3483441

[25] Alsobeh, A., Shatnawi, A.M., Al-Ahmad, B., Aljmal, A., & Khamaiseh, S.Y. (2024). AI-Powered AOP: Enhancing Runtime Monitoring with Large Language Models and Statistical Learning. International Journal of Advanced Computer Science and Applications. DOI: 10.14569/IJACSA.2024.0151113

[26] Muehlburger, H., & Wotawa, F. (2024). FaultLines - Evaluating the Efficacy of Open-Source Large Language Models for Fault Detection in Cyber-Physical Systems*. *2024 IEEE International Conference on Artificial Intelligence Testing (AITest)*, 47-54. https://doi.org/10.1109/AITest62860.2024.00014