Large Language Model for Smart Inverter Cyber-Attack Detection via Textual Analysis of Volt/VAR Commands

Alaa Selim, *Student Member, IEEE*, Junbo Zhao, *Senior Member, IEEE*, and Bo Yang

***Abstract*—This letter demonstrates a proof-of-concept validation of the Large Language Model (LLM) for smart inverter cyberattack detection through textual control commands. The proposed method can detect the manipulation of Volt/VAR curves and the comparison results with state-of-the-art machine learning techniques highlight its efficacy in identifying cyber attacks. Test results obtained from a real distribution feeder in Colorado, USA, validate the high accuracy for attack classification as well as demonstrate the potential of LLMs in adding a robust security layer for cyber-attacks.**

***Index Terms*—Large language models, cyber-attack detection, Volt/VAR control, text security, distribution network.**

I. Introduction

THE ADVANCEMENT of smart grid and communication technologies has induced more potential cyber vulnerabilities. For example, the Volt/VAR control commands used by smart inverters if compromised by an attacker can lead to over or under-voltage issues [1], [2]. Most existing attack detection techniques, such as [3], rely predominantly on numerical data, which may not provide sufficient context for identifying cyber threats effectively. Transforming numerical Volt/VAR control commands into text can enhance security by masking direct numerical details and increasing the interpretability of the data. Such a format takes advantage of operators' expertise, facilitating a more instinctive recognition of subtle irregularities and bolstering defenses against cyberattacks in grid operations.

LLMs like GPT-4 and Machine Learning (ML) classifiers like XGBoost serve different purposes and have distinct advantages depending on the application. When it comes to tasks such as detecting normal and attack commands in a smart grid, here's how an LLM might have advantages over machine learning models: a) **Adaptability to New Data**: LLMs can continue learning from new data and fine-tuned, improving their understanding and detection capabilities over time [4]. While ML models can also be updated, they might require retraining with a curated dataset and may not adapt as seamlessly to new types of language-based patterns. b) **Contextual Analysis**: LLMs can analyze the context surrounding a command, such as the time of issuance, the sequence of previous commands, and the current state of the grid. This allows for a more nuanced detection of whether a command is likely normal or an attack. c) **Explainability and Reasoning**: LLMs can often provide reasons for their predictions, explaining why a particular command might be considered suspicious. This can be valuable for human operators who need to understand the rationale behind a decision [5].

This paper introduces a new approach by employing Large Language Models (LLMs) – a tool predominantly used in natural language processing – for the detection of smart inverter cyber-attacks. By analyzing the patterns and anomalies in Volt/VAR commands, this paper aims to showcase the potential of LLMs in enhancing the cybersecurity of smart grids. Additionally, the potential of LLMs for future enhancements, like having timestamps and user authentication, opens avenues for precision improvement.

II. Problem Statement

The attacks on Volt/Var curves of smart inverters focus on simulating an attack through stealthy modifications to a set of voltage points in Volt/VAR curve, denoted as $\mathbf{V}_0$. The attack subtly alters these nominal voltage points to create a new set, $\mathbf{N}$, while adhering to defined constraints. The stealthy changes are implemented as follows:

$$\mathbf{N}[i] = \begin{cases} \text{clip}(\mathbf{V}[i] + \delta_i, \mathbf{V}_0[i] - \beta, \mathbf{V}_0[i] + \beta), & \text{if } i = 0, \\ \max(\text{clip}(\mathbf{V}[i] + \delta_i, \\ \quad \mathbf{V}_0[i] - \beta, \mathbf{V}_0[i] + \beta), \mathbf{N}[i-1] + h), & \text{if } i > 0, \end{cases} \tag{1}$$

where $\delta \sim \mathcal{N}(0, \alpha)$ represents the attack-induced alterations. Here, $\mathbf{V}_0$ denotes the original set of voltage points. Parameters α and β define the range of alteration and clipping threshold, respectively. Additionally, h is the minimum required change between successive measurements, ensuring gradual and realistic transitions. The attack simulation adheres to the control rules for Volt/VAR curves in IEEE 1547 standard [1]:

$$v_r - 0.18 \leq v_1 \leq v_2 - 0.02, \tag{2}$$
$$v_r - 0.03 \leq v_2 \leq v_r, \tag{3}$$
$$v_r \leq v_3 \leq v_r + 0.03, \tag{4}$$
$$v_r + 0.02 \leq v_4 \leq v_r + 0.18, \tag{5}$$
$$-\bar{q} \leq q \leq \bar{q}, \tag{6}$$
$$0.95 \leq v_r \leq 1.05, \tag{7}$$
$$|\Delta v_n| \leq \alpha. \tag{8}$$

The Volt/VAR curve shown in Fig. 1 is characterized by voltage points $(v_l, v_1, v_2, v_r, v_3, v_4, v_h)$ and reactive power points $(-\bar{q}, \bar{q})$. We adopt the assumption in [2] that the Volt/VAR curve is odd symmetric around the axis $v = \bar{v} = v_r$, where v_r denotes the reference voltage around which the system is regulated. The voltage points v_l, v_1, v_2, v_3, v_4 and v_h represent specific operational voltage limits within the control scheme. The reactive power is indicated by q, with $\bar{q}$ representing its maximum limit. The symbol Δv_n refers

Manuscript received 21 January 2024; revised 27 May 2024 and 12 August 2024; accepted 27 August 2024. Date of publication 2 September 2024; date of current version 23 October 2024. This work was supported in part by the U.S. Department of Energy's Office of Cybersecurity, Energy Security, and Emergency Response, and in part by Hitachi America Ltd. Paper no. PESL-00027-2024. *(Corresponding author: Junbo Zhao.)*

Alaa Selim and Junbo Zhao are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269 USA (e-mail: alaa.selim@uconn.edu; junbo@uconn.edu).

Bo Yang is with the Research and Development Division, Hitachi America Ltd., Santa Clara, CA 95054 USA (e-mail: bo.yang@hal.hitachi.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSG.2024.3453648.

Digital Object Identifier 10.1109/TSG.2024.3453648

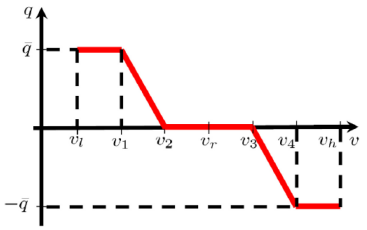


Fig. 1. Standard Volt/VAR curve [1].

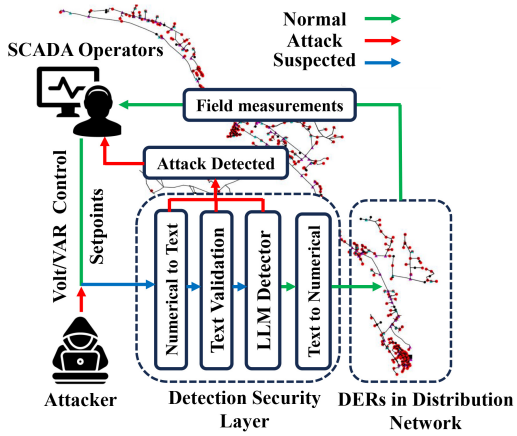


Fig. 2. Conceptual model of the proposed detector using LLM.

to the voltage deviation for any of the voltage points above, and α signifies the maximum allowable deviation of voltage points due to cyber-attack in (1). In our attack model, $\mathbf{V}[i]$ and $\mathbf{V}_0[i]$ are vectors that represent the dynamic and baseline voltage coordinates at the i-th control round, respectively. Specifically, $\mathbf{V}[i] = (v_l[i], v_1[i], v_2[i], v_r[i], v_3[i], v_4[i], v_h[i])$ and $\mathbf{V}_0[i] = (v_l^0[i], v_1^0[i], v_2^0[i], v_r^0[i], v_3^0[i], v_4^0[i], v_h^0[i])$, where $v_x[i]$ are the actual voltage measurements at specific points on the Volt/VAR curve, and $v_x^0[i]$ are their expected or nominal values under normal operational conditions.

III. Cyber-Attack Detection Using LLM

The proposed LLM-based detection framework is shown in Fig. 2. An important step is to convert the numerical Volt/VAR control commands into text format as shown in the Appendix, following the utility company's standard text format, and then validate them to ensure free from potential malicious attacks. The data is then analyzed by a fine-tuned LLM detector for various attack scenarios. When any potential attack is detected, the operator will receive an alert, and the command is blocked. Otherwise, the command is converted to its numerical form and executed through the smart inverters. Thus, the detection security layer is comprised of three main interconnected stages, each of which is detailed below.

A. Numerical to Text Conversion

Our approach converts traditional numerical commands in Volt/VAR systems into text-based commands, facilitating their use in text-based models like LLMs and GPT-4. It is important to note that while attackers may have access to the input layer of control commands, our model is designed to handle text that could potentially contain attack vectors. The classifier is trained to classify any suspicious text commands. As a result, conducting adversarial attacks becomes significantly more challenging because attackers lack detailed information about the classifier model and its decisions. Given a set of initial values $I = \{i_1, i_2, \ldots, i_n\}$ corresponding to variables $X = \{X_1, X_2, \ldots, X_n\}$, and a dataset D containing observations $\{d_{1,1}, d_{1,2}, \ldots, d_{1,n}, \ldots, d_{m,1}, \ldots, d_{m,n}\}$ for each variable across m rounds, the change Δ for each data point in a round is computed.

$$\Delta_{j,k} = d_{j,k} - i_k, \tag{9}$$

where $\Delta_{j,k}$ is the change for the k^{th} variable in the j^{th} round, $d_{j,k}$ is the data point in the j^{th} control round for the k^{th} variable, and i_k is the initial value for the k^{th} variable. The calculated change $\Delta_{j,k}$ is then converted into a text command. For each variable X_k in round j, the command is defined as:

$$\text{Command}_{j,k} = \begin{cases} \text{“Shift } X_k \text{ left by } |\Delta_{j,k}| \text{ units”} & \text{if } \Delta_{j,k} < 0 \\ \text{“Shift } X_k \text{ right by } |\Delta_{j,k}| \text{ units”} & \text{if } \Delta_{j,k} \geq 0 \end{cases} \tag{10}$$

Each round of j then compiles these commands into a sequence representing the changes across all variables X. The set $X = \{X_1, X_2, \ldots, X_n\}$ is mapped to control the predefined nominal voltage vector $(v_l, v_1, v_2, v_r, v_3, v_4, v_h)$ in the previous section.

B. Data Validation Phase

Let S denote the set of all strings, and $D \subset S$ the set of strings adhering to the prescribed data format. D is defined over two lines as:

$$D = \{s \in S \mid s = \text{“Shift X”}_i \text{ “ ” } d \text{ “ by 0.0”}_{xx} \text{ “ units”}, \tag{11}$$

$$i \in \{1, \ldots, 6\}, d \in \{\text{“right”}, \text{“left”}\}, xx \in [0, 999]\} \tag{12}$$

The validation function $V : S \to \{0, 1\}$ is then given by:

$$V(s) = \begin{cases} 1, \text{ if } s \in D, \\ 0, \text{ otherwise.} \end{cases} \tag{13}$$

This function asserts the adherence of a string s to the data format of the utility, returning 1 for format compliance and 0 otherwise.

C. Adapting Language Model for Cyber-Attack Classification

1) BERT Base Architecture Adaptation: We utilize the BERT model [6] as the LLM for cyber-attack classification. Its self-attention mechanism is given by

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{14}$$

where Q, K, and V are queries, keys, and values, respectively, and d_k is the key dimension. This is essential for capturing contextual relationships in cyber-security texts and it allows for a comprehensive understanding of cyber-attack patterns.

2) Efficiency Through DistilBERT: DistilBERT employs knowledge distillation from BERT, crucial for real-time cyber-attack detection. The distillation process uses a loss function combining supervised learning loss and distillation loss:

$$L = \alpha \times L_{\text{CE}}(y, \sigma(z_s)) + (1 - \alpha) \times T^2 \times KL(\sigma(z_s/T), \sigma(z_t/T)), \tag{15}$$

where L_{CE} is cross-entropy loss, KL is Kullback-Leibler divergence, σ is the softmax function, z_s and z_t are student and teacher logits, y represents true labels, T is a temperature parameter, and α balances the loss components. This tailored training makes DistilBERT efficient for cyber-attack detection.

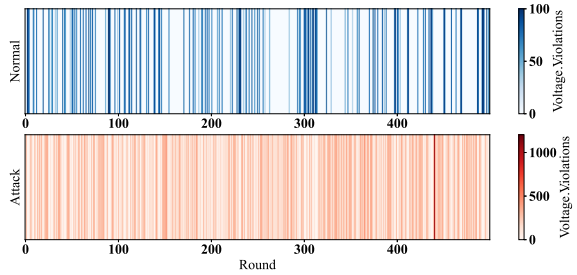


Fig. 3. Dataset sample for cyber-attack and normal scenarios.

3) Tokenization for Cyber-Security Texts: Tokenization converts cyber-security texts into a structured format. Each word w in the input text is mapped to an index in a vocabulary V, and a special token [*PAD*] is added if necessary. The tokenization function τ maps the input text T to its tokenized form T':

$$T' = \tau(T), \tag{16}$$

where the token segmentation, mapping, and potential truncation to a maximum length L are performed. This process ensures accurate interpretation of technical cyber-security language by the model.

4) Low-Rank Adaptation (LoRA) for Targeted Model Tuning: LoRA [7] adapts the DistilBERT model for cyber-attack classification. It introduces low-rank matrices $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{r\times h}$ to the weight matrix $W \in \mathbb{R}^{d\times h}$ of the transformer module:

$$W' = W + AB, \tag{17}$$

where $\Delta W = AB$ is the low-rank update. During training, only A and B are updated, focusing the model's learning on cyber-security features:

$$\min_{A,B} \mathcal{L}(W + AB). \tag{18}$$

5) Training Process for Cyber-Attack Detection: The training process involves optimizing parameters θ, including those in LoRA's matrices, to minimize the loss function $\mathcal{L}$:

$$\theta^* = \arg\min_{\theta} \mathcal{L}(\theta). \tag{19}$$

Backpropagation and gradient descent are used to update these parameters:

$$\theta_i^{(t+1)} = \theta_i^{(t)} - \eta \frac{\partial \mathcal{L}}{\partial \theta_i}, \tag{20}$$

where η is the learning rate. Regularization techniques like dropout are used to enhance generalization, crucial for effective and reliable cyber-attack classification.

IV. Test Results

The proposed method is tested using an actual 5625-node distribution network in Colorado, USA. We initiate thousands of parallel attack rounds on 30 smart inverters on their Volt/VAR curve settings, causing system voltage violations to increase above the normal case of 100 violations obtained by the snapshot solution of OpenDSS [8]. The nominal voltage vector $(v_l, v_1, v_2, v_3, v_4, v_h)$ for the Volt/VAR curve settings is initialized as (0.5, 0.92, 0.95, 1.0, 1.08, 1.50), which acts as the standard operational baseline. A sample of normal and attack scenarios within the control commands is illustrated in Fig. 3. This figure demonstrates the resultant voltage violations attributable to these attacks, highlighting their stealthiness. Notably, these attacks can induce up to 1,000 voltage violations with a minimal alteration δ of less than 0.05 to the Volt/VAR curve points. These rounds of attacks follow the methodology in (1) and adhere to the constraints of (2) to (8) to make it a realistic attack for the Volt/VAR curve settings inside the inverter. Our detector classifies control commands to Volt/VAR settings that result in violations below 100 as normal or optimized, while those exceeding this threshold are flagged for further examination by our detection security layer in Fig. 2. This threshold of 100 is established by conducting multiple power flow simulations to calculate the maximum number of voltage violations in the system during normal operation. The training of LLM is configured with a set of optimized hyper-parameters. A learning rate η of 2×10^{-6} is employed, with training and evaluation batch sizes of 24 and 16, respectively. The model is trained over 25 epochs, incorporating a weight decay of 0.05 to combat over-fitting. A dropout rate of 0.1 is used for regularization, and α of 0.01 is established for the cyber-attack limit. Given the imbalanced nature of our dataset, the evaluation phase will focus on analyzing precision, recall, and F1-score metrics:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{21}$$

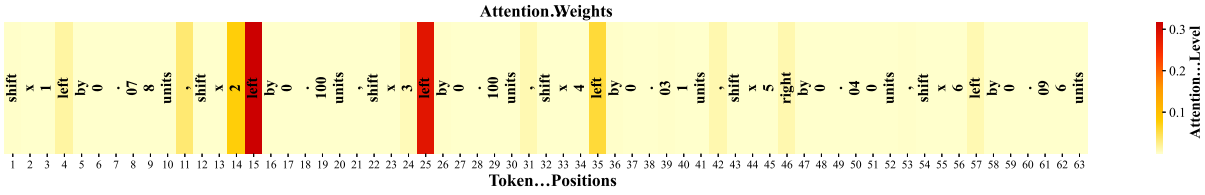
$$\text{Recall} = \frac{TP}{TP + FN}, \tag{22}$$

$$F1\text{-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{23}$$

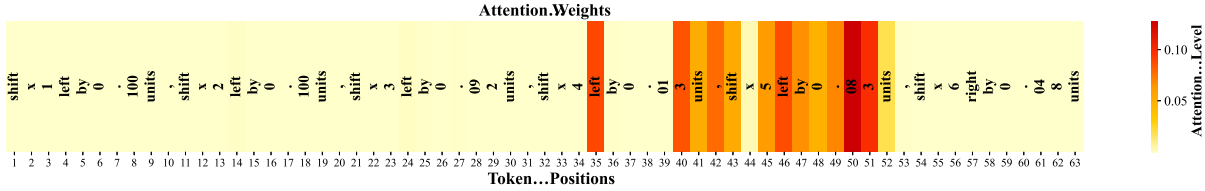
We position our proposed LLM to classify text data, contrasting with other ML classifiers in [9] that deal with numerical data in an imbalanced cyber-attack dataset. Despite this difference, the LLM's performance stands out as shown in Table I. Logistic Regression, working with numerical data, shows modest success, evidenced by weighted average scores of 0.74 in precision, recall, and F1-Score, highlighting challenges in generalizing across the dataset. Perceptron and Naive Bayes, also focus on numerical inputs, and demonstrate intermediate effectiveness with an F1-Score of 0.82, indicating competent but limited capabilities in managing class imbalance. The SVM classifier and others, such as AdaBoost, KNN, Decision Tree, RandomForest, and XGBoost, process numerical data and achieve high F1-Scores (0.94 to 0.97), reflecting their strong performance in the imbalanced dataset. Note that the proposed LLM, designed for text data, matches these results, maintaining a consistent 0.97 across weighted average precision, recall, and F1-Score. It is worth noting that the XGBoost is used here to validate our model's performance. As a result, the performance comparisons with deep learning methods, such as CNN, LSTM, and GRU [10] are carried out to demonstrate the robustness of our proposed LLM. According to the data summarized in Table I, the proposed classifier consistently outperforms these methods across various metrics. Specifically, our model achieves precision, recall, and F1-scores ranging from 0.94 to 0.98, exceeding the scores of CNN, LSTM, and GRU which vary between 0.91 and 0.97. Our LLM model processes decisions within an average inference time of 0.2 seconds on a 64-bit computer with an Intel Core i9-12900KF 3.19 GHz CPU, 128 GB RAM, and NVIDIA GeForce RTX 3090 24 GB GPU. These comparisons between our LLM and other state-of-the-art-based cyber-threat detection methods aim to validate the reliability and performance of the proposed model. Despite the complexity of our text-based data, we

TABLE I
CYBER-ATTACK CLASSIFIER PERFORMANCE METRICS

	Class 0 (Non-attack)			Class 1 (Attack)			Weighted Average		
Classifier	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
Logistic Regression	0.60	0.42	0.50	0.80	0.89	0.84	0.74	0.76	0.74
Perceptron	0.70	0.66	0.68	0.87	0.89	0.88	0.82	0.82	0.82
Naive Bayes	0.72	0.62	0.66	0.86	0.90	0.88	0.82	0.82	0.82
SVM	0.75	0.92	0.82	0.97	0.88	0.92	0.90	0.89	0.89
AdaBoost	0.93	0.87	0.90	0.95	0.97	0.96	0.94	0.94	0.94
KNN	0.91	0.92	0.92	0.97	0.97	0.97	0.95	0.95	0.95
Decision Tree	0.93	0.94	0.94	0.98	0.97	0.98	0.97	0.97	0.97
RandomForest	0.94	0.96	0.95	0.98	0.98	0.98	0.97	0.97	0.97
XGBoost	0.94	0.95	0.95	0.98	0.98	0.98	0.97	0.97	0.97
CNN	0.94	0.93	0.94	0.97	0.98	0.97	0.96	0.96	0.96
LSTM	0.93	0.91	0.92	0.97	0.97	0.97	0.96	0.96	0.96
GRU	0.93	0.92	0.92	0.97	0.97	0.97	0.96	0.96	0.96
Proposed LLM	**0.96**	**0.94**	**0.95**	**0.98**	**0.98**	**0.98**	**0.97**	**0.97**	**0.97**



(a) Normal scenario.



(b) Cyber-attack scenario.

Fig. 4. Demonstrating the explainability of the input text control command using LLM attention weights.

have achieved similar results. Our classifier's dual security layers, unique to textual analysis, preemptively screen for malicious content, a feature absent in ML classifiers. Additionally, the potential of LLMs for future enhancements, like having timestamps and user authentication, opens avenues for precision improvement. Finally, Fig. 4 demonstrates the explainability of our proposed LLM classifier. It visualizes the attention weights across different tokens within a command, where the intensity of the color correlates with the significance assigned by the model. This allows operators to quickly identify critical parts of a command leading to this classification.

V. CONCLUSION

This letter introduces a novel LLM for detecting cyber-attacks in smart grid control systems using textual control commands. This innovative approach not only demonstrates higher classification accuracy compared to traditional methods but also adds a unique security layer to protect Volt/VAR commands. Numerical simulation results on a real feeder in Colorado, USA, yield insightful and practical results. Future efforts will aim to refine our model using advanced LLM techniques and broaden its application in cybersecurity for distribution systems.

REFERENCES

[1] *IEEE Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems Interfaces*, IEEE Standard 1547-2018, 2018, Accessed: May 1, 2024. [Online]. Available: https://standards.ieee.org/ieee/1547/5915/

[2] I. Murzakhanov, S. Gupta, S. Chatzivasileiadis, and V. Kekatos, "Optimal design of volt/VAR control rules for inverter-interfaced distributed energy resources," *IEEE Trans. Smart Grid*, vol. 15, no. 1, pp. 312–323, Jan. 2024.

[3] B. Ahn et al., "An overview of cyber-resilient smart inverters based on practical attack models," *IEEE Trans. Power Electron.*, vol. 39, no. 4, pp. 4657–4673, Apr. 2024.

[4] M. Bakker et al., "Fine-tuning language models to find agreement among humans with diverse preferences," in *Proc. 36th Adv. Neural Inf. Process. Syst.*, 2022, pp. 38176–38189.

[5] K. Valmeekam, M. Marquez, A. Olmo, S. Sreedharan, and S. Kambhampati, "PlanBench: An extensible benchmark for evaluating large language models on planning and reasoning about change," 2022, *arXiv:2206.10498*.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019, *arXiv:1810.04805*.

[7] E. J. Hu et al., "LoRA: Low-rank adaptation of large language models," 2021, *arXiv:2106.09685*.

[8] D. Montenegro, M. Hernandez, and G. A. Ramos, "Real time OpenDSS framework for distribution systems simulation and analysis," in *Proc. 6th IEEE/PES Transm. Distrib. Latin Am. Conf. Expo. (T D-LA)*, 2012, pp. 1–5.

[9] F. Pedregosa et al., "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, no. 85, pp. 2825–2830, 2011.

[10] A. Gulli and S. Pal, *Deep Learning with Keras*. Birmingham, U.K.: Packt Publ. Ltd., 2017.