Unveiling Network Intrusions: Leveraging LargeLanguage Models for Anomaly Detection in Cybersecurity

Vraj Patel, [a] Bhargavi Savani, [b] Bela Shah, [c] and Amit Thakkar [d]

Dept. of Computer Science and Engineering, CSPIT,
Charotar University of Science and Technology,
Anand, India

[a] vrajp1706@gmail.com
[b] bhargavisavani1301@gmail.com
[c] belashah.cs@charusat.ac.in
[d] Corresponding author: amitthakkar.it@charusat.ac.in

Abstract. Large Language Models (LLMs) are recognized for their flexibility which has benefited a variety of domains. One such area of study is log analysis for cybersecurity where LLMs can be a great asset. This research delves deep into utilization of Large Language Models (LLMs), namely RoBERTa and MegatronBERT, to analyze network vulnerabilities, with a specific focus on Hadoop Distributed File System (HDFS) logs. Through training these Large Language Models (LLMs) on this dataset, the work aims to compare the accuracy of identifying outliers. The approach includes preprocessing the data, extracting relevant features, and the utilization of progressive LLM architectures for pattern recognition. By performing a number of experiments, the study demonstrates the effectiveness of LLMs in enhancing network security, offering a promising avenue for proactive defense strategies in modern IT environments.

INTRODUCTION

In the contemporary world, interconnected systems create vast online networks, exemplified by organizational networks that operate continuously. These networks generate extensive logs integral to cybersecurity, offering insights into potential security breaches. Historically, natural language processing (NLP) primarily relied on models such as recurrent neural networks (RNNs) and long short-term memory (LSTM) networks were utilized for analyzing sequential data. However, these models faced limitations such as unidirectional processing, which restricted context consideration to preceding elements during probability calculations for the current word. Additionally, they encountered challenges like gradient vanishing and computational inefficiency when handling lengthy sequences.

The advent of Language Models with Transformers (LLMs) introduced a paradigm shift in NLP. LLMs, featuring a transformer architecture and self-attention mechanism, revolutionized sequential data processing by enabling independent, parallel computations at each sequence position. This mechanism enhances the model's ability to automatically discern and prioritize crucial information, optimizing performance and generalization capabilities. Furthermore, LLMs excel in handling long sequences and capturing long-term dependencies, facilitated by their dynamic weight adjustments and parallel processing capabilities.

The transformer architecture incorporates techniques like residual connections and normalization to mitigate gradient vanishing issues and enhance model training efficacy. This research aims to evaluate and compare two robust LLM systems for anomaly detection using NLP techniques on log data. Moreover, it involves fine-tuning advanced algorithmic models to analyze large log volumes and identify irregular patterns, contributing to advancements in cybersecurity through data-driven approaches.

LITERATURE REVIEW

This section provides a summary of research works related to the field of Large Language Models on Log datasets, along with information on advancements, challenges and achievements.

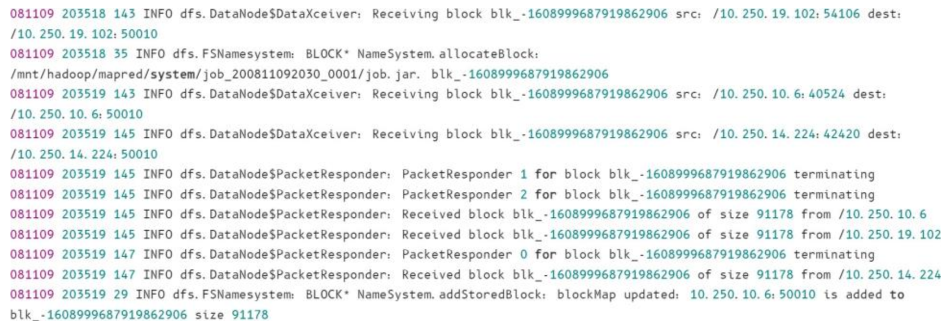
Boffa, Matteo, et al. [1] introduce a comprehensive design process that results in LogPrécis. The authors demonstrate how LogPrécis can facilitate the examination of large datasets containing approximately 400,000 distinct Unix shell assaults. Instead of dealing with hundreds of thousands of individual instances, LogPrécis condenses the data to around 3,000 fingerprints. The effectiveness of LogPrécis in handling large volumes of raw shell session data, automatically identifying attack strategies, reducing data complexity to manageable finger- prints, and grouping sessions based on common attack patterns is highlighted in this study. This approach makes it easier to identify and respond to security threats effectively due to the streamlining the examination and analysis of Unix shell assaults. They have demonstrated that misclassification may result from this architecture. This is the case when adversaries are present and intention ally create attacks that imitate the same fingerprint to get around the system. Karlsen, Egil, et al. [2] investigates LLMs with various architectures, including GPT-2, GPT-Neo, RoBERTa, DistilRoBERTa, and BERT. The fine-tuned sequence classification model RoBERTa achieves a F1-Score, averaging at 0.997, was consistent across six datasets collected from web application and system log origins. This work showcases the efficacy of LLMs in the focus of security log analysis. To facilitate this research, the authors introduce a novel experimentation pipeline called LLM4Sec. The development of the LLM4Sec pipeline and the successful fine-tuning of these models emphasizes the potential of advanced language models in enhancing cybersecurity efforts and detecting potential threats within log data. This analysis indicated that baseline performances, in which training is limited to the classification head, frequently fail to categories any aberrant event, occasionally failing to classify any event at all. Upon examining the false positive rates, it is apparent that the extensive fine-tuning of the underlying Large Language Model severely impairs their capacity to reliably categories sequences for intrusion detection. The model may mistakenly classify anomalous activity as normal, and that is why it is important to balance fine-tuning efforts to avoid overfitting to specific patterns or noise in the data, often mistaking anomalous activity for normal ones.

Hiroki, Kazuo, et al. [3] have suggested using Latent Dirichlet Allocation (LDA), which is a topic modelling technique which is employed to construct a substantial vulnerability database which constitutes active attack scenarios. The authors have specifically referenced two papers, "LOCAL PRIVILEGE ESCALATION" and "BROWSE HACKING". However, the efficiency is compromised if the suggested method is evaluated on a single comparison target. This draw- back signifies that while detecting vulnerabilities from a diverse dataset effective result can be expected, however the effectiveness tends to vary when juxtaposed against a specific attack scenario. The research's findings suggest that the accuracy of collating Local Privilege Escalation (LPE) records was less accurate than those of collating Browser Hacking (BH) materials. Siddhartha, Ashutosh. et al. [4] introducing an innovative framework known as Vulnerabilities and Weaknesses to Common Attack Pattern Mapping (VWC-MAP). They use natural language processing (NLP) techniques to automatically identify all relevant attack techniques of a vulnerability via weakness based on their text descriptions. They have demonstrated a unique two-tiered categorization strategy, the first tier categorized vulnerabilities to weakness and vulnerability to attack strategies in the second tier. Additionally, they provide two brand-new automated methods that use Text-to-Text and link prediction algorithms to map attack methodologies to vulnerabilities. The first method uses a Text-to-Text algorithm to directly map attack methodologies to vulnerabilities. The second method involves the use of link prediction algorithms to map connections between attack patterns. To support this research work, the authoritative cybersecurity databases used are Common Vulnerabilities and Exposures (CVE), Common Weakness Enumeration (CWE), and Common Attack Pattern Enumeration and Classification (CAPEC). the study underscores that increasing the contextual knowledge of cybersecurity words or definitions can help to enhance the accuracy of mapping. One effective approach as asserted by this work is by training big language models (like BERT and T5) on relevant large text corpuses related to cybersecurity.

METHODOLOGY

Dataset

The Hadoop Distributed File System (HDFS) Log Dataset [5, 6] is a comprehensive collection of log data generated by almost 200 Amazon EC2 nodes, which is the primary distributed storage system used by Apache Hadoop. This dataset captures various events, operations, and metadata changes occurring within the HDFS infrastructure, providing valuable insights into system performance, utilization patterns, and potential issues. The HDFS dataset consist of 11,175,629 log events, each linked to a block ID. These log entries are grouped into various log windows based on their block IDs, representing program executions within the HDFS system. For each execution, labels are assigned to denote the presence of anomalies. Within this dataset, there are 16,838 blocks of logs indicating system anomalies.



```
081109 203518 143 INFO dfs.DataNode$DataXceiver: Receiving block blk_-1608999687919862906 src: /10.250.19.102:54106 dest:
/10.250.19.102:50010
081109 203518 35 INFO dfs.FSNamesystem: BLOCK* NameSystem.allocateBlock:
/mnt/hadoop/mapred/system/job_200811092030_0001/job.jar. blk_-1608999687919862906
081109 203519 143 INFO dfs.DataNode$DataXceiver: Receiving block blk_-1608999687919862906 src: /10.250.10.6:40524 dest:
/10.250.10.6:50010
081109 203519 145 INFO dfs.DataNode$DataXceiver: Receiving block blk_-1608999687919862906 src: /10.250.14.224:42420 dest:
/10.250.14.224:50010
081109 203519 145 INFO dfs.DataNode$PacketResponder: PacketResponder 1 for block blk_-1608999687919862906 terminating
081109 203519 145 INFO dfs.DataNode$PacketResponder: PacketResponder 2 for block blk_-1608999687919862906 terminating
081109 203519 145 INFO dfs.DataNode$PacketResponder: Received block blk_-1608999687919862906 of size 91178 from /10.250.10.6
081109 203519 145 INFO dfs.DataNode$PacketResponder: Received block blk_-1608999687919862906 of size 91178 from /10.250.19.102
081109 203519 147 INFO dfs.DataNode$PacketResponder: PacketResponder 0 for block blk_-1608999687919862906 terminating
081109 203519 147 INFO dfs.DataNode$PacketResponder: Received block blk_-1608999687919862906 of size 91178 from /10.250.14.224
081109 203519 29 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated: 10.250.10.6:50010 is added to
blk_-1608999687919862906 size 91178
```

FIGURE 1. Referral logs from dataset

Implementation Details

Data undergoes preprocessing steps, including parsing, mapping, tokenization, and feature extraction, to ready it for input into Language Model-based Learning (LLM) systems. This preprocessing transforms the raw log data into a format suitable for subsequent modeling tasks.

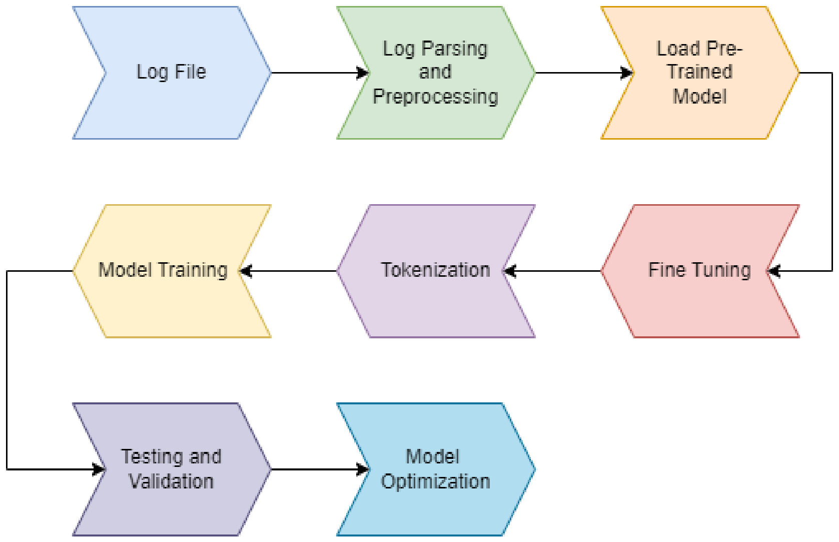


FIGURE 2. Flow Diagram

The process starts with collecting log files from various network devices and systems. These logs contain information about network activities, such as user actions, system events, and network traffic. The collected log files need to be parsed and preprocessed to extract relevant information. This involves tasks such as tokenization, removing unnecessary data, and structuring the logs into a format suitable for further analysis.

Pretrained versions of RoBERTa [7] and MegatronBERT [8, 9], two state-of-the-art Large Language Models (LLMs), are loaded. These models have been previously trained on large corpora of text data and possess general language understanding capabilities. This pretrained RoBERTa and MegatronBERT models are fine-tuned for the specific task of network security anomaly detection using the parsed and preprocessed log data. Fine-tuning involves adjusting the model parameters to adapt it to the characteristics of the target task and dataset.

The preprocessed log data is tokenized using the tokenizer associated with the chosen LLM (RoBERTa or MegatronBERT). Tokenization involves breaking down the text into individual tokens or words, which are then mapped to numerical representations suitable for input into the model. This tokenized log data, along with labels indicating normal and anomalous network behavior, is used to train the fine-tuned LLMs. During training, the model learns to identify patterns and features in the log data that correspond to normal network activity and anomalies.

Once the models are trained, they are evaluated on separate testing and validation datasets to assess their performance. Testing involves applying the models to unseen log data and measuring their ability to accurately detect anomalies. Validation is used to fine-tune hyperparameters and ensure the model generalizes well to new data. After testing and validation, the models may undergo optimization techniques to improve their performance further. This could involve techniques such as hyperparameter tuning to enhance the model's accuracy and robustness.

System Configuration

The all-different model was trained using an integrated GPU chip. The configuration of the device includes a 2 TB SATA Enterprise HDD with an NVIDIA RTX A5000 graphics card, and two Intel Xeon 16 core CPUs with a total of 128 GB DDR4 RAM. Model training and code implementation are done with Jupyter Notebook version 6.4.3.

EVALUATION MATRIX

Table 1. Details of evaluation metrics where TP is True Positive, TN is True Negative, FP is False Positive and FN is False Negative

S. No.	Performance Metric	Definitions	Formulae
i	Accuracy	How correctly a model predicts or classifies instances in a dataset.	$A = \frac{TP + TN}{TP + TN + FP + FN}$
ii	Precision	It is ratio of positive identifiers that are actually correct.	$P = \frac{TP}{TP + FP}$
iii	Recall	It is ratio of actual positive that are identified correctly.	$R = \frac{TP}{TP + FN}$
iv	F1-score	A metric used in binary classification tasks to assess the balance between precision and recall	$F1 = 2 * \frac{P * R}{P + R}$

RESULT ANALYSIS

The experiment results provide insights into the performance of different model architectures (RoBERTa, and MegatronBERT) with different strategies for network security anomaly detection. The experimental results before and after fine tuning are given in Table 1 and Table 2 respectively. The goal is to have the utmost accuracy possible. The highest accuracy achieved for the dataset is 88.97% in MegatronBERT and for RoBERTa, the accuracy achieved is 74.39%. Therefore, MegatronBERT was selected as the best model for this dataset. The findings of this evaluation emphasize the effectiveness of LLMs for log anomaly detection. The loss, accuracy, and F1 score obtained demonstrate the robustness and reliability of the LLM in accurately identifying and classifying logs. Furthermore, the results highlight the importance of Fine-tuning LLMs based on the type of data being analyzed.

Table 2. Experimental results before fine-tuning

Model Architecture	**Loss**	**Accuracy (in %)**	**F1-Score (in %)**
RoBERTa	0.6826	70.63	65.74
MegatronBERT	0.5360	72.00	74.08

Table 3. Experimental results after fine-tuning

Model Architecture	**Loss**	**Accuracy (in %)**	**F1-Score (in %)**
RoBERTa	0.6682	74.39	67.11
MegatronBERT	0.3049	88.97	78.19

CONCLUSION

In conclusion, the utilization of Large Language Models (LLMs) such as RoBERTa and MegatronBERT holds significant promise for enhancing network security through anomaly detection in HDFS logs. However, as compared to RoBERTa, MegatronBERT demonstrates remarkable capabilities in processing vast amounts of unstructured log data, enabling more accurate identification of suspicious activities and potential threats within network traffic.

By leveraging their deep learning architectures, these LLMs can effectively learn complex patterns and anomalies, contributing to the proactive monitoring and defense against cyber threats. However, the successful implementation of these models relies on robust training datasets, efficient feature extraction techniques, and continual adaptation to evolving cybersecurity landscapes. Further research and development are essential to optimize their performance and scalability for real-world deployment in securing network infrastructures.

Future extensions for Large Language Models (LLMs) such as RoBERTa and MegatronBERT in network security for anomaly detection using HDFS logs could involve enhancing their ability to comprehend and contextualize complex log data patterns. This could include developing more robust unsupervised learning techniques to detect anomalies in log sequences, leveraging self-attention mechanisms to capture long-range dependencies and subtle correlations within log data, and exploring techniques to incorporate domain specific knowledge or embeddings to improve model understanding of network security contexts. Additionally, further research could focus on optimizing model architectures and training strategies to handle the scale and variability of HDFS log data efficiently, as well as investigating methods for continual learning to adapt to evolving network environments and emerging threats effectively.

REFERENCES

1. Boffa, Matteo, et al. "LogPrécis: Unleashing Language Models for Automated Shell Log Analysis." arXiv preprint arXiv:2307.08309 (2023).
2. Karlsen, Egil, et al. "Benchmarking Large Language Models for Log Analysis, Security, and Interpretation." arXiv preprint arXiv:2311.14519 (2023).
3. H. Koyanagi, K. Takaragi, S. Wohlgemuth and K. Umezawa, "Threat Analysis Using Topic Models in Large-Scale Vulnerability Databases and Security Incident Case Documents," 2021 IEEE International Symposium on Technologies for Homeland Security (HST), Boston, MA, USA, 2021, pp. 1-6, doi: 10.1109/HST53381.2021.9619846.
4. S. S. Das, A. Dutta, S. Purohit, E. Serra, M. Halappanavar and A. Pothen, "Towards Automatic Mapping of Vulnerabilities to Attack Patterns using Large Language Models," 2022 IEEE International Symposium on Technologies for Homeland Security (HST), Boston, MA, USA, 2022, pp. 1-7, doi: 10.1109/HST56032.2022.10025459.
5. Wei Xu, Ling Huang, Armando Fox, David Patterson, Michael Jordan. Detecting Large-Scale System Problems by Mining Console Logs, in Proc. of the 22nd ACM Symposium on Operating Systems Principles (SOSP), 2009.
6. Jieming Zhu, Shilin He, Pinjia He, Jinyang Liu, Michael R. Lyu. Loghub: A Large Collection of System Log Datasets for AI-driven Log Analytics. IEEE International Symposium on Software Reliability Engineering (ISSRE), 2023.
7. Liu, Yinhan, et al. "Roberta: A robustly optimized bert pretraining approach." arXiv preprint arXiv:1907.11692 (2019).
8. Shoeybi, Mohammad, et al. "Megatron-lm: Training multi-billion parameter language models using model parallelism." arXiv preprint arXiv:1909.08053 (2019).
9. NVIDIA. (2019). Megatron-LM. GitHub. https://github.com/NVIDIA/Megatron- LM