

NewsGuard-LLM: A Lightweight, Dynamic LLM based Framework for Real-Time Threat Detection in Digital News Stream

Gaurav Jadhav*, Jagadesh Rajendran*, Amit Kumar Singh*, Zeba Khanam†, Robert Hercock†

*School of Computer Science and Electronic Engineering, University of Essex, United Kingdom

†Centre of Excellence for Cyber Threat Detection, BT Security Research, United Kingdom

Abstract—The accelerated evolution of global digital news ecosystems, driven by increased connectivity and the proliferation of networked platforms, has placed digital news as a high-volume open-source asset for threat intelligence. This development has intensified the need for real-time threat detection in critical sectors, allowing organizations to anticipate and mitigate emerging risks more effectively. Although large language models (LLMs) offer advanced capabilities in natural language understanding and content analysis, their widespread operational deployment remains limited by substantial computational demands, reliance on external cloud infrastructures, and growing concerns over data privacy and governance. To this end, this paper presents a lightweight and scalable LLM framework designed for real-time threat detection in digital news headlines. Unlike traditional LLM architectures, the proposed system operates entirely within the organization’s infrastructure, ensuring data privacy and minimizing external resource dependencies. The framework automates end-to-end pipeline, from data collection and labeling to dynamic retraining and deployment, leveraging GPU acceleration to enhance computational efficiency. The system integrates open-source LLMs for initial data annotation, a dynamic feedback loop for continuous model refinement, and a user-friendly dashboard for real-time threat monitoring. The power and cost analysis of the framework demonstrate that GPU-based training, while drawing higher instantaneous power (433W), completes tasks up to four times faster than CPU-based training, resulting in lower overall energy consumption (0.108 kWh per training run) and reduced cost per task (\$0.0141 vs. \$0.0166). The experimental evaluation of the proposed framework shows a classification accuracy of 96% and 70% reduction in training time.

Index Terms—Dynamic threat detection, news analytics, natural language processing (NLP), large language models (LLMs), GPU acceleration, continuous learning, cybersecurity.

I. INTRODUCTION

The rapid growth of real-time digital news has reshaped how organizations monitor and respond to emerging risks and security threats. As global information flows accelerate, timely identification of threat signals within large-scale news streams has become critical for domains such as cybersecurity, public safety, and critical infrastructure protection. However, the volume, velocity, and evolving nature of modern threats increasingly challenge traditional detection approaches.

Large Language Models (LLMs) such as GPT-4, BERT, and open-source variants have advanced natural language

understanding across applications including content moderation and cyber threat analysis. Despite their capabilities, the high computational cost, reliance on cloud infrastructure, and data privacy concerns associated with full-scale LLMs limit their suitability for real-time, operational deployment. These models also remain vulnerable to adversarial manipulation, data drift, and rapidly evolving linguistic patterns.

The dynamic and adversarial threat landscape encompassing misinformation, cyber incidents, and public safety risks necessitates adaptable and privacy-preserving detection frameworks. Conventional machine learning methods, including logistic regression, support vector machines, and rule-based filters, struggle to capture contextual nuance and evolving language, often resulting in missed threat signals or elevated false-positive rates.

To address these challenges, we propose a lightweight, end-to-end framework for dynamic threat detection in news that combines GPU accelerated training with efficient natural language processing while preserving privacy and resource efficiency. The framework automates the entire pipeline from scalable news ingestion and NLP-based preprocessing to hybrid LLM assisted labeling and dynamic retraining with human feedback-enabling continuous adaptation to evolving data and threat patterns.

The system further integrates GPU acceleration to significantly reduce training overhead and provides a user-centric dashboard for real time threat monitoring and transparency. By operating entirely on-premise and minimizing reliance on external cloud infrastructure, the framework aligns with organizational privacy, compliance, and operational constraints.

II. RELATED WORK

A. NLP for Threat Detection in News and Social Media

Natural Language Processing (NLP) has become a core technology for automated threat detection within digital news, social media, and broader cybersecurity applications. Early frameworks relied on simple keyword and rule-based matching, which struggled to adapt to evolving threat languages and adversarial tactics [9]. Recent studies have leveraged advanced machine learning and deep learning models—such as BERT, RoBERTa, and custom neural networks—to ex-

tract, classify, and profile emerging cyber threats from large-scale unstructured data including news articles, threat reports, forums, and social media posts [3]–[6]. These methods utilize NLP techniques like named entity recognition (NER), topic modeling (e.g., LDA), sentiment analysis, and contextual embedding to enhance detection capabilities and construct dynamic threat intelligence.

Fine-tuning large pre-trained language models on domain-specific threat corpora notably improves detection performance, allowing models to capture nuanced malicious intent and coded language. Ensemble strategies—such as model stacking or voting—combine traditional and deep learning models for increased robustness and accuracy against evolving threat expressions [6], [10]. Furthermore, domain adaptation and transfer learning approaches are employed to improve generalization across platforms and dataset sources [6], [12].

B. Continuous and Lifelong Learning in Threat Detection

Evolving threat landscapes require models that adapt to non-static data distributions and shifting adversary tactics. Continual learning techniques enable NLP models to incrementally learn from new data without catastrophic forgetting of prior knowledge, supporting tasks such as real-time threat assessment on rapidly changing news or social media streams [11], [19], [24]. Strategies such as online learning, active learning, and hybrid human-in-the-loop systems are used to maintain up-to-date detection capabilities and adapt models to domain drift or unforeseen attack types [11], [13], [19].

C. Scalable and Automated Threat Detection Architectures

The transition from static analysis to real-time scalable frameworks is pivotal in operational threat intelligence. Modern systems are characterized by automated data ingestion pipelines, dynamic retraining, and multi-source integration—pulling data from news, social media, and network logs [20], [23]. Cloud-native and edge computing advancements, together with scalable data storage solutions, allow distributed processing and near instantaneous reaction to emerging threats [23]. Integration with Security Information and Event Management systems (SIEMs), Security Orchestration, Automation, and Response platforms (SOARs), and Threat Intelligence Platforms (TIPs) enables holistic coverage, rapid response, and minimized human intervention [23].

D. GPU Acceleration for Model Training and Inference

Scaling machine learning and deep learning for large datasets and real-time detection is facilitated by GPU acceleration, drastically reducing computational times and increasing throughput compared to CPU-bound approaches [9]. Frameworks like TensorFlow and PyTorch, together with libraries such as CUDA and cuDNN, empower practical deployment of complex models—such as transformer networks and ensemble classifiers—on contemporary hardware, including edge devices and cloud environments [9]. Fast inference is vital for applications like live news event monitoring,

where latency and power efficiency are of critical concern. Prior work on low-power, high-performance embedded System-on-Chips (SoCs) identifies security challenges driven by energy constraints and architectural heterogeneity, motivating lightweight and energy-aware protection mechanisms. [29]

E. Privacy Concerns with NLP and LLMs in Threat Detection

The proliferation of large language models (LLMs) introduces acute privacy and data protection challenges—such as inadvertent exposure of personally identifiable information (PII), issues of data provenance, and difficulties in fulfilling data deletion or modification requests. Research identifies strategies such as differential privacy, federated learning, and robust data minimization to mitigate leakage and unauthorized access risks [23]. Regulatory and ethical compliance adds complexity in deployment, especially in sectors handling sensitive news, legal, or governmental intelligence. Recent work has shown that large language models are vulnerable to prompt injection attacks, where adversarial inputs manipulate model behavior. Graph-based learning approaches have been proposed to capture relational prompt structures, enabling more robust detection of such attacks compared to purely sequential text analysis. [27]

F. Key Advances and Remaining Challenges

Although NLP, deep learning, and hardware acceleration have significantly advanced automated threat detection, research continues to address challenges such as handling extreme class imbalance and rare threat types, explainability of model predictions and regulatory compliance, continuous learning to counteract evolving attacker strategies, cross-platform and multilingual capability for global monitoring. Recent studies emphasize that security must be co-designed with mixed-criticality, energy efficiency, and reliability to address the complexity of modern embedded systems [28].

Existing works also aim to integrate explainable AI, advanced privacy-preserving mechanisms, and fully automated feedback-driven retraining pipelines, fulfilling the scalability, efficiency, and trustworthiness requirements of operational threat intelligence [22], [30].

Several challenges remain in operational threat detection. Model robustness and generalization across diverse media sources and languages continue to be limited, particularly in unseen or low-resource contexts. Real-time deployment further introduces latency and throughput constraints under high-volume data streams. Balancing privacy preservation with transparency and explainability remains challenging, while rapid adaptation to emerging threat narratives is constrained by delays in labeling and feedback. Finally, the lack of standardized benchmarks and reproducible evaluation protocols hinders consistent comparison across threat detection approaches.

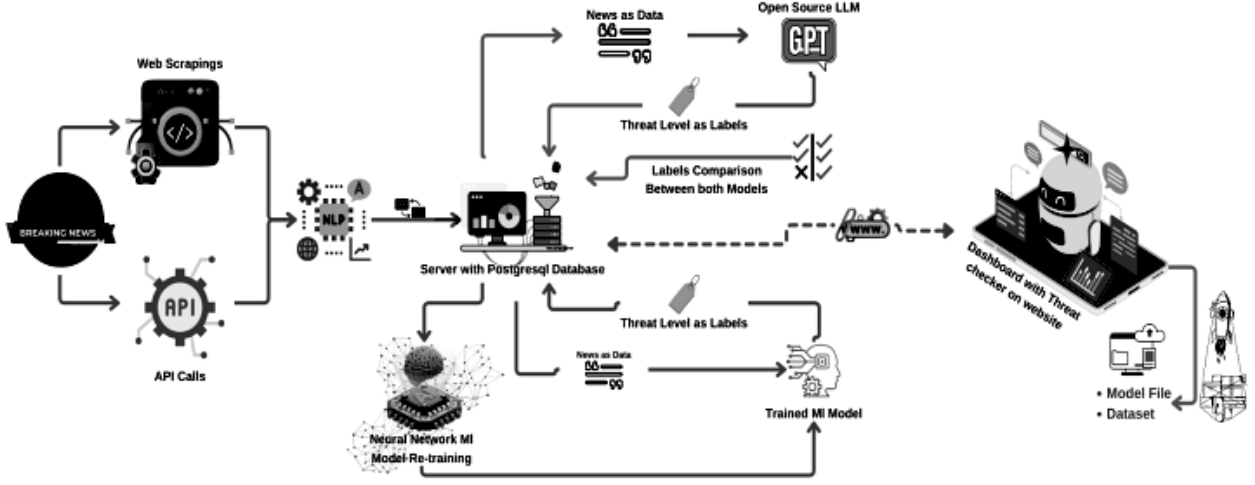


Fig. 1. System Architecture

III. PROPOSED FRAMEWORK

The proposed framework is designed to be both comprehensive and modular, facilitating robust end-to-end automation while maintaining adaptability and privacy preservation. This section details each layer of the technical architecture—from raw data collection to model deployment and real-time monitoring—highlighting the interplay between natural language processing, machine learning, automation, and system engineering.

A. System Overview

Our system operates as a pipeline-based system, structured around the following core modules: (1) Data Collection and Storage, (2) Pre-processing, (3) Hybrid Labeling (Automated and Manual), (4) Model Training and Dynamic Retraining, (5) GPU-Accelerated Infrastructure, (6) Model Deployment, and (7) Dashboard and Visualization. Figure 1 provides an architectural overview, illustrating component interactions and data flows.

Each pipeline module is decoupled, supporting independent updates, parallelization, and integration with external systems via RESTful APIs. Workflow automation is achieved using DAG-based orchestrators (e.g., Apache Airflow), ensuring repeatability, error handling, and logging.

B. Data Collection and Storage

1) *Multi-Source Acquisition*: News headlines and associated metadata are ingested from a diverse set of sources to cover a broad threat landscape and improve the robustness of the model. News APIs (e.g., NewsAPI, GDELT) are used for structured retrieval with customizable keywords and timeframes. Custom web scraping tools such as BeautifulSoup and Selenium collect from curated news portals. Scraping routines are scheduled and monitored for data integrity. Open-source datasets for integration of preannotated

research corpora for benchmarking and bootstrapping. RSS feeds were considered due to their structured and near real-time nature. API-based retrieval was prioritized for richer metadata, filtering flexibility, and historical querying, with RSS ingestion retained as a future extension.

2) *Database Architecture*: Data are ingested into a centralized, high availability PostgreSQL database. The schema is optimized for rapid querying and supports tables for raw data, preprocessed text, labels, model metadata, and audit logs. Connection pooling and backup routines ensure scalability and fault tolerance.

C. Data Preprocessing

1) *Text Processing Pipeline*: This step transforms raw text data into a format suitable for analysis and machine learning. It ensures data quality, consistency, and relevance through a series of Natural Language Processing (NLP) techniques. The following steps are automated using robust NLP libraries such as spaCy, NLTK, WordNetLemmatizer, langdetect, langid or custom scripts.

- *Text Normalization*: It standardizes text data for uniformity, reducing inconsistencies. The tasks include convert text to lowercase (e.g., "Cybersecurity" becomes "cybersecurity"), removing special characters, punctuation, and extra whitespace, and expanding contractions (e.g., "don't" to "do not").
- *Tokenization & Lemmatization*: It converts text into base tokens. For example, by considering a word as a token, the input "Cybersecurity threats are increasing worldwide." will be converted to output ["Cybersecurity", "threats", "are", "increasing", "worldwide"].
- *Stopword Removal*: It filters out common words lacking semantic weight. For example, it eliminates common words like "and," "is," and "the" to focus on meaningful

content, so the output will become ["Cybersecurity", "threats", "increasing", "worldwide"], i.e. "are" is removed.

- **Lemmatization:** It reduces words to their base forms while retaining grammatical meaning in order to reduce sparsity and enhancing the performance of the downstream model. For example, input ["Cybersecurity", "threats", "are", "increasing"] is converted to output ["cybersecurity", "threat", "be", "increase"].
- **Language Detection:** Filters non-English headlines using langdetect.
- **Noise Removal:** It elimination of URLs, HTML tags, special characters, and non-content text. For example, input "Visit https://cybersecurity.com for updates! ;b;Stay safe!;b;" results in output: "visit for updates stay safe".
- **Hashing:** Hash-based and fuzzy matching eliminate redundancies.

Processed data are batch-stored for auditing and re-usability. It is stored in a table within PostgreSQL database.

2) *Preprocessing at Scale:* All preprocessing jobs are parallelized using multiprocessing, and code is containerized for consistent execution across environments. Processing logs and statistics are output for monitoring and debug purposes.

D. Hybrid Labeling Workflow

1) *Automated LLM-Assisted Labeling:* To expedite large-scale annotation, initial threat labels are generated using fine-tuned or open-source LLMs (e.g., OpenLLaMA, BERT) locally, as shown in Fig. 1. An inference pipeline batches headlines, passing them through the LLM to obtain 'High Threat', 'Low Threat', or 'No Threat' labels as well as associated confidence scores. Open-source models including OpenLLaMA, Mistral, and BERT-based classifiers were used for automated labeling. Default inference settings were retained to ensure stability and reproducibility, as parameter variation showed limited downstream impact relative to data quality.

2) *Manual Expert Validation:* A web dashboard displays a representative sample of auto-labeled headlines (stratified by confidence and category) to be approved by human annotators. They validate or correct LLM predictions, Supply additional context (e.g., region, domain) and flag ambiguous or novel cases for further review.

3) *Iterative Feedback Loops:* Validated samples are filtered back into the labeling set. Feedback is also used to fine-tune LLMs or custom classifiers and continually update the labeled corpus for downstream model retraining. Iteration frequency is controlled by performance monitoring and operational needs.

The labeled data is stored in the PostgreSQL database (as shown in Fig. 1) for further processing.

4) *Labeling Taxonomy, Annotators, and Drift Handling:* The labeling taxonomy consists of three classes: High Threat, Low Threat, and No Threat. Automated labels are reviewed using stratified sampling based on confidence scores, and

validated samples are reintegrated into the training corpus. Retraining is periodically triggered based on performance monitoring and data drift indicators.

E. Model Training and Dynamic Retraining

1) *Model Architecture:* Our system employs a lightweight, fully connected neural network (multi-layer perceptron) as its principal text classifier. The architecture is selected based on Low-latency inference, GPU optimization, Minimal memory footprint. Text features are extracted using TF-IDF, FastText, or transformer embeddings from sentence-level models.

2) *Training Pipeline:* Labeled data is split into train/validation/test. Oversampling and class-weight adjustment address strong class imbalances. Dynamic retraining is performed where the model is regularly updated as new labeled threats are ingested or feedback is received (concept drift adaptation).

F. GPU-Accelerated Infrastructure

To achieve GPU acceleration, training is conducted using CUDA-enabled GPUs (NVIDIA A10) with frameworks like PyTorch or TensorFlow. On-GPU augmentation (mixed precision, parallel data loaders) further minimizes training time. Grid/random search tunes layer count, dropout, learning rate, and regularization for hyperparameter optimization.

G. Model Deployment

It deploys its models through containerized environments (e.g., Docker), enabling smooth, consistent rollout across cloud, on-premise, or edge hardware. RESTful APIs built with Flask or FastAPI expose model endpoints for real-time and batch inference, supporting seamless integration with news pipelines and analyst tools. Security is prioritized via authenticated endpoints, encrypted communications, and audit logging. Automated orchestration with Apache Airflow schedules retraining and monitoring, while Kubernetes manages resource scaling and reliability, quickly distributing model updates as needed.

H. Dashboard and Visualisation

Dashboard offers an interactive and user-friendly interface. It is built using frameworks like Dash or Plotly and visualizes key metrics such as accuracy, class counts, and system trends in real time while providing notification for critical threats. Users can monitor performance, review predictions, and explore data via intuitive visual tools like confusion matrices and threat timelines. The dashboard also supports human-in-the-loop feedback, allowing analysts to correct labels and annotate headlines, with full audit logging and access controls to ensure compliance and data integrity.

IV. EXPERIMENTAL EVALUATION

This section provides a comprehensive examination of the experimental setup, dataset characteristics, model performance, comparison of hardware configurations, analytical insights, and the effectiveness of the proposed framework through multifaceted quantitative and qualitative evaluations.

TABLE I
HEADLINE ANALYSIS METRICS

Metric	Value
Total Raw Headlines	1629000
Unique, Clean Headlines	1412000
High Threat Headlines	60000
Low Threat Headlines	1352000
High Threat Proportion	3.7

A. Setup Overview

For training and inference, the CPU used is Intel Xeon with 16GB of RAM, where 75% of 16GB RAM has been utilised. The GPU used is NVIDIA A10 with 24GB RAM, where 80% of the memeory has been utilised.

B. Dataset Overview

Over 1.6 million news headlines were collected from diverse, authoritative news sources using both APIs (e.g., NewsAPI, GDELT) and custom web scraping, ensuring high domain coverage and temporal diversity. Table I shows a overview of this dataset. Rigorous cleaning steps included normalization, tokenization, lemmatization, stopword and noise removal, deduplication, and strict language filtering to ensure quality and consistency. After cleaning and deduplication, 1,412,000 unique English-language headlines remained-providing a rich foundation for robust machine learning. Approximately 3.7% of headlines were labeled as “High Threat,” with the remainder categorized as “Low Threat,” presenting a significant class imbalance that was addressed via oversampling and class-weighted loss functions.

C. Preprocessing Impact

Data reduction from 1,629,000 to 1,412,000 headlines resulted primarily from the removal of duplicates and non-English content. A notable skew toward low-threat headlines required strategic approaches (random oversampling, class weighting) in model training to ensure sensitivity to minority classes.

D. Feature Engineering

- **TF-IDF:** Term Frequency-Inverse Document Frequency transformation was performed to highlight key terms relevant for threat detection. This enhanced feature representation for traditional models and neural networks.
- **Word Embeddings:** Transformer-based embeddings (e.g., BERT) and FastText vectors were also utilized to capture contextual and semantic nuances.

Model Training and Evaluation Protocol: The GPU used for this is NVIDIA A10 (24GB VRAM), Ubuntu 22.04 LTS with a Intel Xeon (16GB RAM) CPU. A fully connected neural network (multi-layer perceptron) designed for resource efficiency and rapid GPU training. The cleaned dataset was partitioned into training, validation, and test sets (e.g., 70/20/10 split) to ensure unbiased evaluation. Tuned for batch size, learning rate, dropout rates, and epochs, with model checkpoints to prevent overfitting.

TABLE II
CPU AND GPU TRAINING RESULTS

Metric	GPU Value	CPU Value
TF-IDF Transformation Time	78.93 seconds	0.38 seconds
Training Time	907.13 seconds	4,027.93 seconds
Testing Time	11.84 seconds	8.98 seconds
Peak GPU Memory Usage	5.5 GB	9 GB RAM
Model Accuracy	95.93%	95.99
Precision (High)	0.96	0.96
Recall (High)	1.00	1.00
F1-Score (High)	0.98	0.98
Precision (Low)	0.82	0.79
Recall (Low)	0.14	0.16
F1-Score (Low)	0.23	0.27

Baseline models including TF-IDF based Logistic Regression, Support Vector Machines, and sentence level transformer embeddings were evaluated using identical data splits. Traditional linear models showed reduced sensitivity under strong class imbalance, while transformer embeddings increased computational overhead with limited performance gains. The proposed model offers the best trade-off between accuracy, efficiency, and on-prem deployment suitability.

Precision-Recall analysis was preferred over ROC evaluation due to the low prevalence of high-threat samples. The model maintained high precision across recall ranges for the high-threat class, while class-weighted training produced stable and well-calibrated confidence estimates.

E. Performance Results

Table II shows GPU and CPU based results with comparative insights. The model achieved around 96% accuracy, demonstrating robust classification performance in distinguishing high and low-threat news headlines. GPU-based training offers a clear advantage in both training time and memory efficiency, making it the preferred choice for operational scenarios where speed and resource optimization are critical. In contrast, CPU-based training, while considerably slower, provides slightly improved classification for particular labels, such as “Low Threat,” as reflected in the marginally higher F1-score. This trade-off emphasizes the value of GPU acceleration in large-scale and time-sensitive deployments, while also highlighting the nuanced gains accessible with CPU throughput in specific contexts.

F. Power Consumption and Cost Efficiency

TABLE III
CONFIGURATION POWER AND COST METRICS

Configuration	Power (W)	Time (h)	Energy (kWh)	Cost (USD)
GPU	433	0.25	0.10825	0.0141
CPU	84.18	1.5167	0.12766	0.01659

Table III shows power, time, energy and cost results for doing the training on the GPU and CPU. The GPU model has the highest instantaneous power draw (433 W) but only runs for 0.25 hours, resulting in modest total energy use (0.10825 kWh) and low cost per task (\$0.0141). The CPU model

consumes significantly less power (84.18 W) but operates for much longer (1.5167 hours), leading to slightly more total energy (0.12766 kWh) and a slightly higher task cost (\$0.01659). The cost of energy is computed using a unit rate of \$0.13 per kWh, e.g., for GPU it is $= 0.10825 \text{ kWh} \times \$0.13 \text{ per kWh} = \$0.0141$.

Comparison with LLM LLaMA 3.2 7B Model: The considered LLaMA 3.2 model has 7 billion parameters. As per the reported figures, it takes 916,000 GPU hours and 700W of power to train the model. Therefore, total energy is $916,000 \text{ GPU hours} \times 700\text{W} = 641,200 \text{ kWh}$. With a unit rate of \$0.13 per kWh, training cost $= 641,200 \text{ kWh} \times \$0.13 \text{ per kWh} = \$83,356$. This cost is too much in comparison to the cost of \$0.0141 for training on a single GPU considered in our case.

G. Real-Time Testing and Monitoring, and Analytical Insights

The model was continuously evaluated on live news streams (20,000 articles/day), with real-time predictions logged and compared to a manually-validated sample. Dashboard provided visualizations for accuracy, precision, recall, system resource usage, and time-based drift, enabling immediate diagnosis and iteration. Dynamic retraining was automated to incorporate new, manually-validated data, ensuring continuous calibration and adaptation. Maintained 95% accuracy across hardware, platforms, and daily dataset shifts. Continuous monitoring minimized concept drift and maintained detection sensitivity. The user dashboard enabled both technical and non-technical personnel to monitor model status and intervene when necessary.

V. CONCLUSION

We introduced a dynamic, lightweight, and end-to-end machine learning framework designed to tackle the complexities of real-time threat detection in news media using efficient and privacy-preserving techniques. By integrating GPU-accelerated training with sophisticated natural language processing (NLP) methodologies, it offers a resilient alternative to conventional cloud-based LLM systems, particularly in resource-constrained or sensitive operational environments. In future, we plan to extend the technique to have multi-modal (e.g., audio and video as well) threat detection capabilities.

VI. ACKNOWLEDGMENT

This work is partially funded by the EPSRC through US-UK collaborative research seed funding in semiconductor security.

REFERENCES

- [1] Manning, C. D., Raghavan, P., Schütze, H., *Introduction to Information Retrieval*, Cambridge University Press, 2008.
- [2] Zhao, S., Gao, L., Ruan, Y., *Automatic Extraction of Cybersecurity Threats from Unstructured Text*, IEEE International Conference on Big Data, 2019.
- [3] *NLP-based techniques for Cyber Threat Intelligence*, ScienceDirect, [Online]. Available: <https://www.sciencedirect.com>.
- [4] Nurul Aina Qistina Abd Razak, Siti Maryam Zainob, Siti Radziah Muhammad Rizuwan, *Generative AI in NLP: Proposed Techniques For Cybersecurity Vulnerability Detection*, TechRxiv, 2025.
- [5] *What Is Natural Language Processing (NLP)?*, Proofpoint, [Online]. Available: <https://www.proofpoint.com>.
- [6] Somagani Venkatesh, et al., *Automated emerging cyber threat identification and profiling based on NLP*, International Journal Of Progressive Research In Engineering Management And Science, 2025.
- [7] Liu, B., *Sentiment Analysis and Opinion Mining*, Morgan & Claypool Publishers, 2012.
- [8] Joulin, A., Grave, E., Bojanowski, P., Mikolov, T., *Bag of Tricks for Efficient Text Classification*, arXiv preprint arXiv:1607.01759, 2016.
- [9] Banerjee, S., *A Machine Learning Approach for the NLP-Based Analysis of Cyber Threats and Vulnerabilities*, Information Security Journal: A Global Perspective, vol. 29, no. 3, pp. 123–134, 2020. doi:10.1080/19393555.2020.1753959.
- [10] *What is Continual Learning?*, IBM, [Online]. Available: <https://www.ibm.com>.
- [11] Michieli, U., Ozay, M., *HOP to the Next Tasks and Domains for Continual Learning in NLP*, arXiv preprint arXiv:2403.08154, 2024.
- [12] Jaiswal, B. D., *Designing Scalable Software Automation Frameworks for Threat Detection and Response*, International Journal of Scientific Research and Management, 2025.
- [13] Edapurath Vijayan, Naveen, *Real-Time Cyber Threat Detection Using Big Data Analytics*, Journal of Engineering and Applied Sciences Technology, 2024.
- [14] *GDELT Project, Global Database of Events, Language, and Tone*, [Online]. Available: <https://www.gdeltproject.org>.
- [15] *Real-Time Cyber Threat Detection Using Big Data Analytics*, Onlinescientificresearch.com, [Online]. Available: <https://www.onlinescientificresearch.com>.
- [16] *NVIDIA CUDA Toolkit Documentation*, NVIDIA, [Online]. Available: <https://docs.nvidia.com/cuda>.
- [17] *Docker Documentation*, [Online]. Available: <https://docs.docker.com>.
- [18] *How GPUs Supercharge AI and ML for Breakthroughs*, Hyperstack, [Online]. Available: <https://www.hyperstack.cloud>.
- [19] *How GPUs Enhance Machine Learning and AI Performance*, Aethir, [Online]. Available: <https://www.aethir.com>.
- [20] *How GPUs Accelerate Deep Learning*, Gcore, [Online]. Available: <https://www.gcore.com>.
- [21] Paszke, A., et al., *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, Advances in Neural Information Processing Systems, 2019.
- [22] Mohale, V.Z., et al., *A systematic review on the integration of explainable artificial intelligence (XAI) in cybersecurity: Challenges and future directions*, Frontiers in Artificial Intelligence, vol. 8, 2025.
- [23] Yan, B., et al., *On Protecting the Data Privacy of Large Language Models (LLMs): A Survey*, arXiv preprint arXiv:2403.05136, 2024.
- [24] *Privacy Considerations in Large Language Models*, Google Research, [Online]. Available: <https://research.google>.
- [25] Doshi-Velez, F., Kim, B., *Towards a Rigorous Science of Interpretable Machine Learning*, arXiv preprint arXiv:1702.08608, 2017.
- [26] Samek, W., et al., *Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models*, arXiv preprint arXiv:1706.07269, 2017.
- [27] G. Jadhav, A. K. Singh, Z. Khanam and R. Hercocok *Novel GNN-based Approach for Detection of Prompt Injection Attacks*, IEEE International Conference on Cyber Security and Resilience (CSR), 2025.
- [28] S. S. Sahoo et al., *Emergent Design Challenges for Embedded Systems and Paths Forward: Mixed-criticality, Energy, Reliability and Security Perspectives: Special Session Paper*, International Conference on Hardware/Software Codesign and System Synthesis, 2021.
- [29] P. Rahimi, A. K. Singh, X. Wang and A. Prakash, *Trends and Challenges in Ensuring Security for Low-Power and High-Performance Embedded SoCs*, IEEE 14th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoc), 2021.
- [30] Sarker, I.H., *Explainable AI for cybersecurity automation, intelligence and resilience: Principles, challenges and future directions*, Journal of Information Security and Applications, 2024.