SaRLog: Semantic-Aware Robust Log Anomaly Detection via BERT-Augmented Contrastive Learning

Jilcha Lelisa Adeba, Deuk-Hun Kim, and Jin Kwak

***Abstract*— Numerous deep learning-based methods have been developed to address the intricacies of anomaly detection tasks within system logs, presenting two significant challenges. First, balancing model complexity with the capacity to generate semantically meaningful representations for the downstream detection model, is a delicate task. Second, these methods generally depend on extensive labeled data for effective training. Despite efforts to address these challenges separately, a comprehensive solution that efficiently tackles both issues simultaneously are lacking. In response, we introduce SaRLog, a comprehensive solution designed to overcome the limitations of existing methods by leveraging the contextual semantic information extraction capability of bidirectional encoder representations from transformers (BERT) and the few-shot learning capability of the Siamese network. The Siamese network, featured with contractive loss, is implemented on top of a custom domain-specific fine-tuned BERT. Our comparative analysis validates SaRLog's effectiveness against established baseline methods, demonstrating F1 score improvement of up to 31.2% and 46.7% on BGL and Thunderbird datasets respectively. Moreover, additional experimental analysis aimed at evaluating the few-shot learning capability highlights the robustness and generalization efficiency of SaRLog. Thus, by overcoming dataset variability and improving model generalization, SaRLog advances log anomaly detection, thereby effectively handling complex log data challenges.**

***Index Terms*— Anomaly Detection, BERT, Contrastive Loss, IoT, Log Preprocessing, Pretrained Language Model, Siamese Network.**

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. 2021R1A2C2011391) and was supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2021-0-01806, Development of security by design and security management technology in smart factory). *(Corresponding author: Jin Kwak).*

Jilcha Lelisa Adeba is with ISAA Lab., Department of AI Convergence Network, Ajou University, Suwon 16499, South Korea (e-mail: jilchalelisa@ajou.ac.kr).

Deuk-Hun Kim is with ISAA Lab., Institute for Computing and Informatics Research, Ajou University, Suwon 16499, South Korea (e-mail: dhkim.isaa@gmail.com).

Jin Kwak is with Department of Cybersecurity, Ajou University, Suwon 16499, South Korea (e-mail: security@ajou.ac.kr)

Mentions of supplemental materials and animal/human rights statements can be included here.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org

I. INTRODUCTION

WITHIN the contemporary landscape marked by the convergence of the Internet of Things (IoT) and cloud computing [1], log anomaly detection has emerged as pivotal for maintaining the integrity and security of complex, distributed systems [2]. With connected IoT devices projected to number 55.7 billion by 2025 potentially generating approximately 80 zettabytes of data [3], the efficient management, analysis, and interpretation of log data becomes critical to guarantee that interconnected systems operate reliably and securely. Automated log anomaly detection systems, particularly those powered by deep learning technologies, are essential tools, offering a swift and sophisticated means for identifying and addressing potential threats and anomalies within vast networks [4]. This technological synergy enhances the precision and speed at which anomalies are detected, significantly reducing the risk of system failures and cybersecurity threats.

Log messages, often denoted as logs, are semi-structured records of events occurring within a system, application, or device [5], [6]. These records, generated to capture information about the system's behavior, errors, warnings, and other relevant activities, are crucial for detecting security breaches, software errors, system faults, and performance issues [3]. At the core of the log message is an unstructured statement, formulated during the software development process, comprising constant and variable parameters [6]. The constant part discloses the event template, while the variable parts contain parameters that convey dynamic runtime information.

A central aspect shared among existing log anomaly detection methodologies is log parsing [7], [8], [9], a technique that translates each log message into its specific static event template with associated variable parameters. This is followed by the construction of log sequences [8], [10], [11] and the transformation of these sequences into vector representations [11], [12], [13], which are subsequently fed into downstream anomaly detection models. Previous studies predominantly utilized static embedding techniques such as word2vec and FastText for constructing vector representations [8], [11], [14]. However, these methods often neglect the semantic information inherent in raw log messages, thus decreasing the detection system's robustness. Recent studies indicate a shift toward using pretrained language models (PLMs) such as bidirectional encoder representations from transformers (BERT) and generative pretrained transformers

(GPT) for generating representation vectors for the downstream detection task [11], [12], [13]. This shift highlights the importance of capturing semantic contextual information present in log data, a crucial element in improving the performance of log-based anomaly detection systems.

Nonetheless, these off-the-shelf PLMs present certain limitations when applied to domain-specific tasks such as intrusion detection [15], [16]. The primary challenge stems from their limited understanding of specialized terminologies within the domain. Section III discusses this issue in the context of log anomaly detection. Moreover, the diverse distributional and structural characteristics of log data across different datasets (Section III), combined with the challenges of acquiring sufficient training data [17], emphasize the necessity for developing robust detection systems capable of reliably managing both environmental and data drift, thus ensuring their effectiveness across various contexts and datasets. In response, we introduce a contrastive learning-based log anomaly detection method named SaRLog (semantic-aware robust log) anomaly detection, leveraging a domain-specific fine-tuned PLM and a custom Siamese network.

The key contributions of this work lie in the innovative approach to addressing two critical challenges. First, the proposed model effectively balances the model's overall complexity and efficient representation, combining contextual semantic representations' capability of a domain-specific pre-trained BERT-based language model with the few-shot learning capability of a Siamese architecture. Second, the use of contractive loss and a fine-tuned domain-specific representation model enhances adaptability to rare and subtle anomalies, thereby reducing dependence on extensive labeled datasets. Furthermore, through rigorous experimental analysis on the BGL and Thunderbird datasets, SaRLog demonstrated superior performance compared to baseline methods, achieving remarkable F1-scores of 0.9880 and 0.9993, respectively. By addressing the challenges posed by dataset variability and enhancing model generalization, SaRLog contributes to the advancement of anomaly detection solutions capable of effectively navigating the complexities of diverse log data.

The article is structured as follows: Section II reviews prior studies on common approaches in log anomaly detection, covering log parsing, representation, and detection. Section III addresses challenges in pretraining language models on log data and examines the potential advantages of domain specific PLMs. Section IV presents technical details of the proposed model, including preprocessing, context-aware semantic representation, and contrastive learning-based detection. Evaluation results and comparisons with baseline methods are discussed in Section V. Limitations and future research directions are outlined in Section VI. Finally, Section VII offers concluding remarks.

II. Related works

Several existing log-based anomaly detection approaches rely on effective log parsing methods [18], [19], [20], [21] to extract structured event templates from raw log data [7], [8], [9]. IPLOM [18] utilizes hierarchical clustering alongside heuristic strategies, such as grouping by log length and word position, and employing word mapping relationships to enhance parsing accuracy. Similarly, SLCT [19] introduces a clustering algorithm that identifies frequent words in log content, facilitating parsing through frequency-based clustering. Drain [20] employs a method that leverages natural language processing (NLP) to filter out irrelevant variables and constructs parse trees based on the length of log messages, thus enabling structured log template parsing. Spell [21] presents a streaming-based parsing technique using the longest common subsequence, enabling the extraction of structured log templates and their parameters. However, these techniques often rely on predefined templates or patterns to parse logs, making them less adaptable when faced with diverse and evolving log data. Additionally, they may struggle with handling noisy or incomplete logs, which can lead to inaccurate parsing results.

Moreover, log anomaly detection necessitates the efficient representation of log messages, where either raw log messages or parsed log sequences are transformed into meaningful representation vectors. Various methods employing different neural network-based representation techniques, including static and contextual embeddings, have been proposed. Inspired by word2vec, static log representation methods such as logkey2vec [7] and Template2Vec [8] have been widely adopted. However, these methods do not fully exploit the rich contextual semantic information embedded within log messages. Consequently, recent studies [11], [12], [13] have emphasized the use of semantic and context-aware representation models such as GPT, BERT, and RoBERTa.

Conversely, prior studies underscore the significance of capturing temporal dependencies within log messages for downstream detection tasks, given that anomalies are often discerned by analyzing sequences of log events [22]. Various techniques have been developed, including CNN-based [7], [9], [23], RNN-based [8], [23], [10], and attention-based [8], [11], [12], [13] approaches. DeepLog [10] utilizes LSTM to model temporal dependency for log entry-level online anomaly detection. LogAnomaly [8] proposes an attention-based LSTM combined with template2vector. LogRobust [11] employs Bi-LSTM and attention mechanisms to extract bidirectional dependencies. Furthermore, PLELog [14] and LogAT [23] tackle log anomaly detection through semi-supervised and transfer learning, respectively, aiming to alleviate the need for extensive manually labeled data. However, a key limitation of these studies is their reliance on static representation methods, which may compromise the capture of nuanced semantic information, particularly in complex log structures.

Furthermore, NeuralLog [12] and LAnoBERT [13] aim to enhance robustness and generalizability in log anomaly detection utilizing the BERT-based representation approach. However, exploiting contextual semantic information

TABLE I

DISTRIBUTION OF THE TOP 10 WORDS IN BGL AND THUNDERBIRD DATASETS

Rank	BGL normal samples		BGL anomalous samples		Thunderbird normal samples		Thunderbird anomalous samples	
	Vocabulary	Proportion	Vocabulary	Proportion	Vocabulary	Proportion	Vocabulary	Proportion
1	ras	0.1257	ras	0.1122	may	0.1374	opendemux	0.0990
2	kernel	0.1189	fatal	0.1121	kernel	0.0517	may	0.0961
3	info	0.1093	kernel	0.0990	user	0.0383	error	0.0593
4	generating	0.0499	error	0.0700	mosal	0.0330	in	0.0518
5	iar	0.0186	interrupt	0.0697	va	0.0234	pbsmom	0.0495
6	dear	0.0182	data	0.0696	protctx	0.0229	connection	0.0495
7	alignment	0.0171	tlb	0.0492	from	0.0228	refused	0.0495
8	exceptions	0.0171	to	0.0215	cannot	0.0224	cannot	0.0495
9	microseconds	0.0158	on	0.0213	mosalvirttophysex	0.0223	connect	0.0495
10	error	0.0154	message	0.0208	retrieve	0.0223	to	0.0495

embedded in log messages often involves complex models such as transformers for downstream detection tasks [12], aiming to process the high-dimensional vector output of the representation model. Despite the advantages of improved semantic understanding, the risk of memorization is a significant concern when using complex models such as BERT and transformers in combination. Memorization refers to a model learning the specifics and noise in the training data to the extent that it adversely affects the model's performance on new, unseen data, compromising its generalization ability [24]. Hence, in this study, we propose an innovative approach that efficiently addresses the aforementioned challenges.

III. SIGNIFICANCE OF DOMAIN SPECIFIC LANGUAGE MODEL IN LOG ANOMALY DETECTION

Recent studies highlight the widespread adoption of BERT and its variants for generating contextual and semantically meaningful representations for log-based anomaly detection tasks [11], [12], [13]. BERT is a transformer encoder-based language model pretrained on a massive corpus of publicly available text data [25]. Given a sequence of input tokens $\{x_i^T\}_{i=1}^n$, the objective of BERT with masked language modeling is to maximize the probability of predicting the masked tokens x_i^T given the surrounding context $x_{<i}^T$ and $x_{>i}^T$, as shown in (1).

$$P(x) = argMax_\theta \prod_{i=1}^{n} P(x_i | x_{<i}, x_{>i}; \theta) \quad (1)$$

where, θ is the model's parameters and n is sequence length.

Furthermore, BERT employs a multi-head attention mechanism to enhance the training [25]. Given a sequence of input tokens $x^T = \{x_1^T, x_2^T, \ldots, x_n^T\}$, where n is the sequence length, the self-attention mechanism of BERT computes the attention scores between all pairs of tokens using multiple sets of Query (Q), Key (K), and Value (V) matrices, as shown in (2). The self-attention mechanism is subsequently applied h times, where h is the number of attention heads, with different learned projection matrices and the outputs of each attention head are concatenated and linearly transformed to obtain the final vector. This enables BERT to capture diverse contextual information, learn richer representations, mitigate overfitting, and achieve better generalization on downstream NLP tasks such as text classification [26], named entity recognition [27], and sentiment analysis [28].

$$Attention(Q, K.V) = SoftMax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2)$$

where Q is a query matrix with dimension $(n\ x\ d_Q)$, K is a key matrix with dimension $(n\ x\ d_k)$, V is a value matrix with dimension $(n\ x\ d_v)$, and d is the dimensionality of the corresponding vector.

Although BERT and BERT-based PLMs have shown impressive capabilities in various NLP tasks, their application to log-based anomaly detection presents significant challenges [29]. Their primary limitation is that, being inherently generic, they may lack the domain-specific knowledge essential for understanding specialized terminology found in system logs [30]. The unique vocabulary, error codes, and contextually relevant terms typical of system logs may not be fully captured, leading to a suboptimal semantic representation of log entries. For instance, terms such as *"thread," "driver," "kernel," and "key"* may hold different meanings in general English compared with their usage within log messages, illustrating the difficulty in achieving accurate word comprehension within the distinct linguistic context of system logs.

Moreover, system log datasets are characterized by sparsity and comparatively short sentence lengths. Table I demonstrates the distribution of the top ten words in two publicly available datasets, BGL and Thunderbird, following the text cleaning process. The total counts of unique words in BGL for normal and anomaly samples are 769 and 209, respectively, whereas for the Thunderbird dataset, these counts are 3002 and 65. Despite this, the top ten most frequent words in the normal samples of the BGL and Thunderbird datasets account for 50.6% and 39.65% of the total occurrences in their corresponding class for each dataset, respectively. Similarly,

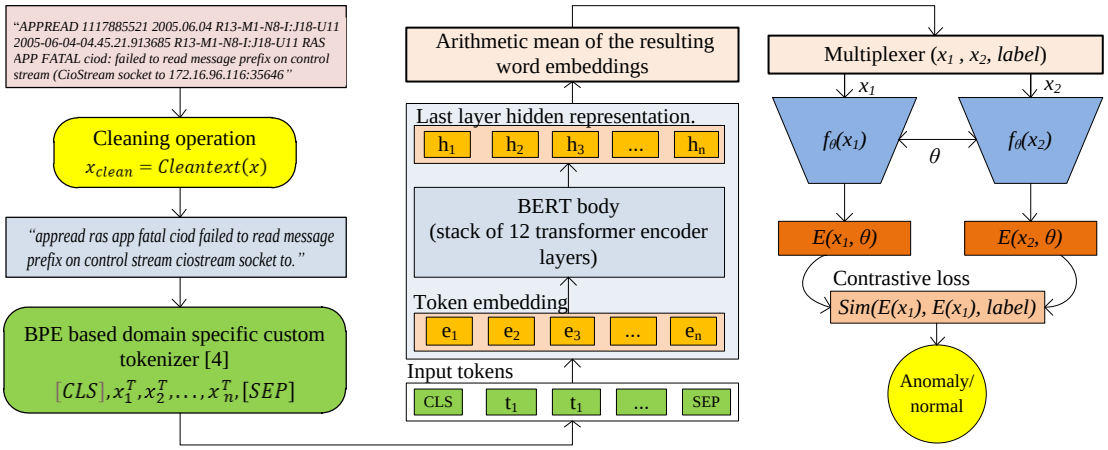


Fig. 1. Overall architecture of the proposed model.

the top ten most frequent words in the anomalous sample of the BGL and Thunderbird datasets constitute 64.54% and 60.32% of the overall anomalous samples of each dataset, respectively. Nonetheless, these datasets show minimal commonality in the most frequent terms, with *"kernel"* being the sole term to appear in the normal samples of both.

The observed statistical information highlights the potential challenges associated with pretraining language models, such as BERT [25], on system log datasets. These challenges fall into two main categories. The first is that the limited context provided by small vocabulary size undermines the model's ability to capture rich contextual information, potentially leading to suboptimal representation. The second challenge is the skewed vocabulary coverage caused by the dominance of a few frequent words, which may limit exposure to less common vocabulary during pretraining, adversely affecting the model's ability to generalize to new words. To address this issue, either fine-tuning models on a large corpus of log datasets from diverse sources or utilizing pretrained models specifically adapted to relevant target domains, such as cybersecurity is necessary. Such approaches can enhance the model's ability to grasp the nuances of language and terminology present in system logs, thereby improving performance and generalization across different log datasets.

IV. Proposed Method (SaRLog)

To enhance anomaly detection in system logs, we present SaRLog, a method that combines domain-specific PLMs for semantic and context-aware feature extraction with a Siamese network employing contrastive loss for effective similarity learning. This section provides an in-depth overview of SaRLog's operation, covering preprocessing, representation, and detection procedures. Fig. 1 illustrates the overall architecture of the proposed approach.

A. Preprocessing

The preprocessing stage encompasses a text cleaning and tokenization phase. During the cleaning phase, raw log messages are sanitized and standardized by removing irrelevant features. The input text x undergoes a series of operations $x_{clean} = Cleantext(x)$ to improve its quality prior to tokenization. This converts all characters to lowercase, removing HTML tags, special characters, punctuation, and numerical values. For example, a raw log message, "*APPREAD 1117885521 2005.06.04 R13-M1-N8-I:J18-U11 2005-06-04-04.45.21.913685 R13-M1-N8-I:J18-U11 RAS APP FATAL ciod: failed to read message prefix on control stream (CioStream socket to 172.16.96.116:35646"*, is transformed into *"appread ras app fatal ciod failed to read message prefix on control stream ciostream socket to."*

The cleaned text, x_{clean}, is then subjected to tokenization, ensuring the input is devoid of noise and primed for the extraction of semantically rich representations. We utilize a custom tokenizer algorithm, as proposed in [15], which segments lengthy words into smaller tokens to better manage out-of-vocabulary (OOV) words. This tokenizer, based on the byte pair encoding (BPE) method [31], is tailored for handling OOV and domain-specific terms within the cybersecurity field. This adaptation is critical for efficient log-based anomaly detection due to potential vocabulary overlaps between system log messages and cybersecurity texts [22]. Terms related to network configurations, user authentication, and system events often indicate security incidents within system logs.

Subsequently, special tokens for classification [CLS] and separation [SEP] will be added to the resulting sequence of tokens. The sequence is then adjusted to a uniform length of 512 through padding or truncation, and an attention mask is created to identify the elements within the sequence that require attention. The final product of the preprocessing stage is a token sequence $x^T = \{[CLS], x_1^T, x_2^T, \ldots, x_n^T, [SEP]\}$, where *T* denotes a token, and *n* represents the sequence length, fixed at 512.

B. Context-Aware Semantic Representation

This step involves mapping the sequence of input tokens to numerical vectors, with a deliberate emphasis on ensuring that these vectors accurately encapsulate the semantic and contextual nuances inherent in the respective tokens. As discussed in Section III, BERT models pretrained on general English text face challenges in accurately capturing the unique linguistic context of system logs [29], [30]. To address this, we employ SecureBERT [15], a domain-specific model that

has been fine-tuned on a substantial corpus of cybersecurity text. SecureBERT features 12 hidden layers, each with an output dimension of 768, 12 attention layers, and a feed-forward network size of 2048, with an input size of 512.

For a given sequence of input tokens $\{x_i^T\}_{i=1}^n$, the objective of the representation model is to produce a contextual and semantical embedding vector $x^e = \{x_1^e, x_2^e, \ldots, x_n^e\} \in R^d$, where e represents embedding, n is the sequence length, and d is the dimensionality of each embedding vector, set at 768. In our study, we specifically leveraged the word embeddings output by the last encoder layer to produce a comprehensive representation, x^r, for a given log message. We adopted a straightforward aggregation method, specifically, calculating the arithmetic mean of the resulting word embeddings $\{x_i^e\}_{i=1}^n$ across all words in the log message, as shown in (3). This aggregation approach enables the distillation of semantic information from individual word embeddings into a unified representation for the entire log message, thereby enhancing the effectiveness of the subsequent anomaly detection task.

$$x^r = \frac{1}{n}\sum_{i=1}^{n} x_i^e \tag{3}$$

where x^r is the final representation vector, n is the sequence length, and x_i^e is the word embedding for the i^{th} word in the log message.

C. Contrastive Learning-based Detection

Given a pair of input vectors x_1^r and x_2^r, encapsulating rich semantic information captured by the representation model, and a Siamese network with a shared trainable parameter θ, the training objective is to optimize θ such that the similarity metric $S(x_1^r, x_1^r)$ identifies patterns indicative of the similarity or dissimilarity between the two vectors. The Siamese network consists of two identical fully connected neural networks. Each network had an input size of 768, corresponding to the embedding dimension of the representation model, and included two hidden layers of sizes 64 and 32 with ReLU activation functions, culminating in an output layer of size 1. For implementation, a single network was constructed and utilized sequentially for both pairs of input data. To avoid confusion, we refer to these as subnetworks 1 and 2, respectively. Each subnetwork is represented by the embedding function $f_\theta(x_1^r) = E(x_1^r, \theta)$ and $f_\theta(x_2^r) = E(x_2^r, \theta)$, where E denotes embedding, and θ denotes the shared trainable parameter, Fig. 1. The similarity metric S computes the geometric relationship between numerical vectors derived from the embedding functions, as shown in (4). Furthermore, a multiplexer module, acting as a pooling layer, was designed that efficiently presents input pairs to the network, along with corresponding labels indicating their similarity or dissimilarity, as shown in Fig. 1.

$$S(x_1^r, x_2^r) = sim(f_\theta(x_1^r), f_\theta(x_2^r)) \tag{4}$$

A contrastive loss function, as formulated in (5), was employed to refine the training objective, optimizing the model to reduce the distance between embeddings of similar inputs while increasing the distance between embeddings of dissimilar inputs. Throughout the training process, the parameter θ is updated concurrently for both subnetworks, ensuring the acquisition of discriminative features. These features enable the network to generate efficient embeddings for both inputs. The Euclidean distance d between the two embeddings is calculated, and the contrastive loss is then determined based on a margin parameter *m* and the Euclidean distance d, as expressed in (5). The margin parameter *m* establishes a threshold delineating the proximity or disparity required between the embeddings of similar and dissimilar samples.

$$L(y, d) = (1 - y) * \frac{1}{2} * d^2 + y * \frac{1}{2} * max(0, m - d)^2 \tag{5}$$

where y is the binary label, 1 for similar inputs with 0 for dissimilar inputs, and d is the Euclidian distance between the embeddings of x_1 and x_2.

The right-hand side of (5) is designed to minimize the distance between embeddings of similar pairs, imposing penalties for larger distances, thus fostering proximity. Conversely, the left side of the equation motivates the embeddings of dissimilar pairs to maintain a separation of at least m. If the distance d exceeds *m*, further adjustment for that pair is unnecessary, as the loss becomes zero. During the training phase, the objective is to reduce the aggregate contrastive loss across all input pairs within the training dataset, as expressed in (6). In the testing phase, each sample from the test dataset is compared with a reference data point from the normal samples. A sample is anomalous if the distance between the two pairs surpasses the predefined margin parameter *m*; otherwise, it is deemed normal.

$$J = \frac{1}{N}\sum_{i=1}^{N} L(y_i, d_i) \tag{6}$$

where *N* is the total number of pairs in the training dataset, with y_i and d_i respectively corresponding to the binary label and the distance for the i^{th} pair.

V. EXPERIMENTAL ANALYSIS

The experimental analysis, performed on the BGL and Thunderbird datasets, showcases the superior performance of the proposed approach (SaRLog) compared to baseline methods. This section provides an in-depth assessment of the model's performance across several dimensions, including the efficacy of the representation model, the model's performance in few-shot learning scenarios, and its comparative performance against baseline models.

A. Experimental Setting

TABLE III

PERFORMANCE OF SALOG AGAINST BASELINE METHODS

Baseline Methods	BGL			Thunderbird		
	Precision	Recall	F1-score	Precision	Recall	F1-score
LogAnomaly [8]	0.972	0.944	0.963	0.611	0.780	0.684
LogRobust [11]	0.625	0.967	0.753	0.616	0.781	0.681
NeuralLog [12]	0.985	0.985	0.985	0.933	**1.000**	0.964
PLELog [14]	0.965	**0.999**	0.982	0.821	0.952	0.881
SaRLog (ours)	**0.994**	0.982	**0.988**	**1.000**	0.999	**0.999**

The experimental analysis utilized Visual Studio Code alongside the PyTorch framework for machine learning tasks and Python 3.11 for coding and experimentation. The experiments were conducted on a high-performance workstation equipped with a 64-bit Ubuntu 22.04.3 LTS OS, powered by an Intel Xeon® Gold 5122 CPU @ 3.60GHz with 8 cores, 128GB of RAM, and an NVIDIA Quadro P5000 GPU. The training phase spanned 10 epochs, incorporating a 20% dropout rate, using the Adam optimizer with a learning rate of 0.01, and binary cross-entropy with logits (BCEWithLogitsLoss) served as the loss function. The task of log anomaly detection was approached as a binary classification problem, with the model's performance being assessed via precision, recall, and the F1-score. $Precision = \frac{TP}{TP + FP}$, $Recall = \frac{TP}{TP + FN}$, $F1 - score = 2\left(\frac{Precision * Recall}{Precision + Recall}\right)$, where TP = true positive, FP = false Positive and FN = false negative. These measures provide a quantifiable evaluation of the proposed model's effectiveness and facilitate a fair comparison with baseline methods.

B. Datasets

The proposed model's performance and its comparison with prior approaches were evaluated using two publicly available datasets: BGL and Thunderbird [6]. This section introduces these datasets and relevant statistical information, contributing to an understanding of their scope and the context of the experimental analysis. Additionally, Table II presents detailed statistical data pertaining to the portions of the dataset utilized for training and testing.

The BGL dataset originates from a supercomputing system and comprises 4,747,963 log messages that were collected by the Lawrence Livermore National Laboratory (LLNL) [6]. Each message within this dataset has been manually classified as either normal or anomalous, with 348,460 identified as anomalous. Similarly, the Thunderbird dataset [6] includes a total of 44,841,030 entries, consisting of 41,592,791 alerts and 3,248,239 non-alerts. This dataset was sourced from the Thunderbird supercomputer system at the Sandia National Laboratories (SNL). The detailed presentation of these datasets provides researchers with a foundation for understanding the experimental setup and the basis for a fair comparison with existing methods.

TABLE II

STATISTICAL INFORMATION OF THE DATASETS

Datasets	Log events	Training data		Testing data	
		Total	Alert	Total	Alert
BGL	1,847	55,401	25,066	13,851	6,309
Thunderbird	2,880	400,000	193,840	100,000	3,460

C. Comparative Analysis: Proposed Model Versus Baseline Methods

We conducted a comparative analysis to evaluate the performance of SaRLog against established baseline methods, with the objective of determining its performance in accomplishing the target task. The proposed model exhibited remarkable achievement across both datasets, achieving an F1-score of 0.988 on BGL and 0.999 on Thunderbird. These outcomes signify substantial enhancements over the baseline methods, marking improvements of up to 31.2% and 46.7% respectively, as shown in Table III.

TABLE IV

PERFORMANCE OF SARLOG WHEN USING DIFFERENT EMBEDDING ARCHITECTURE

Embedding Architecture	Dataset	Precision	Recall	F1-score
Word2vec	BGL	0.842	0.977	0.885
	Thunderbird	0.801	0.960	0.873
GPT-2	BGL	0.985	0.910	0.946
	Thunderbird	0.992	0.952	0.971
BERT	BGL	0.955	0.970	0.963
	Thunderbird	0.993	0.999	0.995
RoBERTa	BGL	0.965	0.981	0.973
	Thunderbird	1.000	0.999	0.999
SecureBERT	BGL	**0.994**	**0.982**	**0.988**
	Thunderbird	1.000	0.999	0.999

Although NeuralLog [12] achieves a performance level comparable to SaRLog with an F1-score of 0.985 on the BGL dataset, it marginally lags on the Thunderbird dataset, with an F1-score of 0.964, which is 3.5% lower. LogAnomaly [8] and LogRobust [11] exhibit considerable fluctuations in performance between the two datasets. Specifically, LogAnomaly's F1-score shows a sharp decline of 28.9% when transitioning from BGL to Thunderbird, whereas LogRobust's F1-score maintains relative stability, witnessing only a 9.6% reduction, albeit starting from a lower performance

benchmark. Furthermore, PLELog [14] lags behind SARLog on both datasets. SARLog shows a 0.6% F1-score improvement over PLELog for the BGL dataset and a 13.4% improvement for the Thunderbird dataset. These numerical differences highlight the proposed method's robustness and dependability in anomaly detection tasks across diverse datasets.

D. Performance of the Detection Head

In our exploration of different architectures for anomaly detection, we observed a remarkable difference in training loss convergence rates between the Siamese network and a multilayer fully connected network, both configured with an equivalent level of model complexity regarding trainable parameters. Specifically, the Siamese network attained convergence within the initial four epochs, as depicted in Fig. 2 (b), whereas the fully connected network required at least

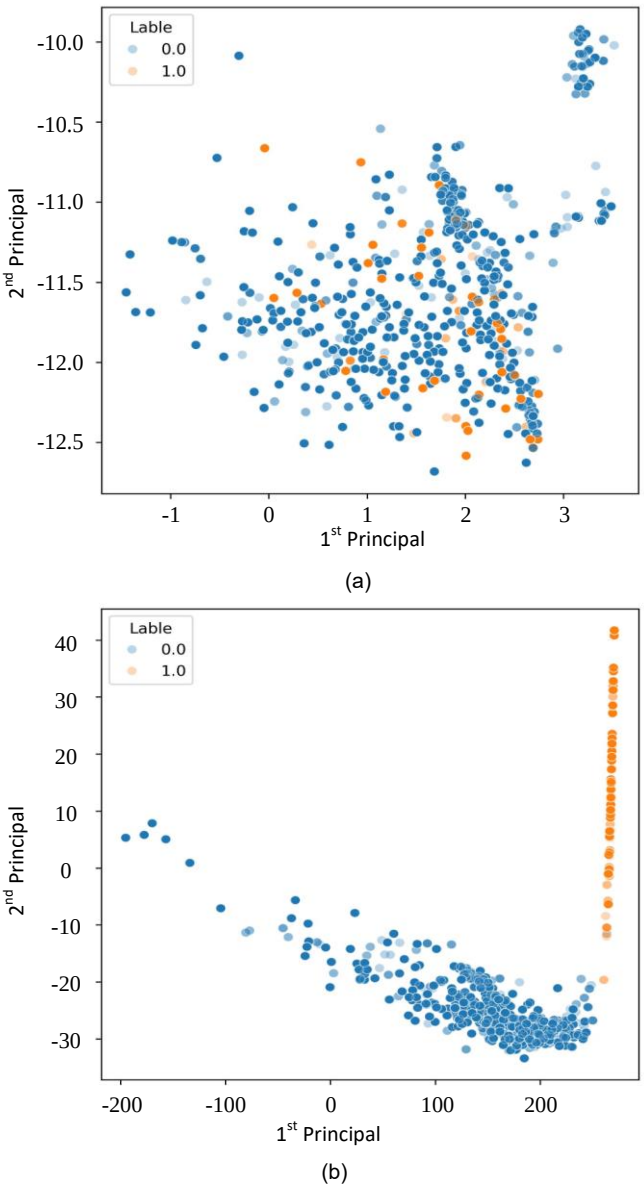


Fig. 3. PCA illustration of the representation vector. BERT hidden layer (a) and output layer of Siamese network (b).

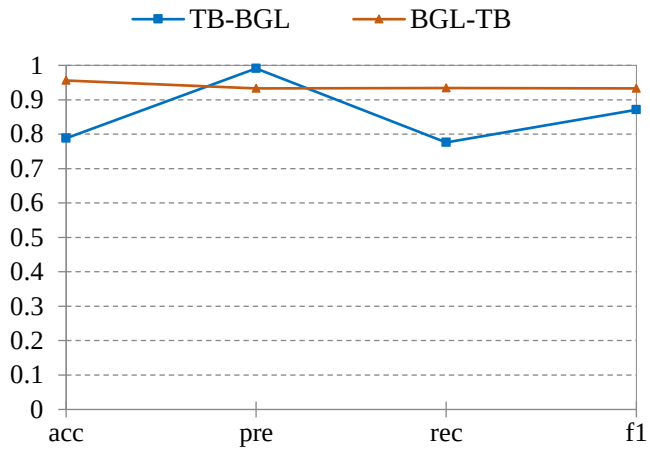


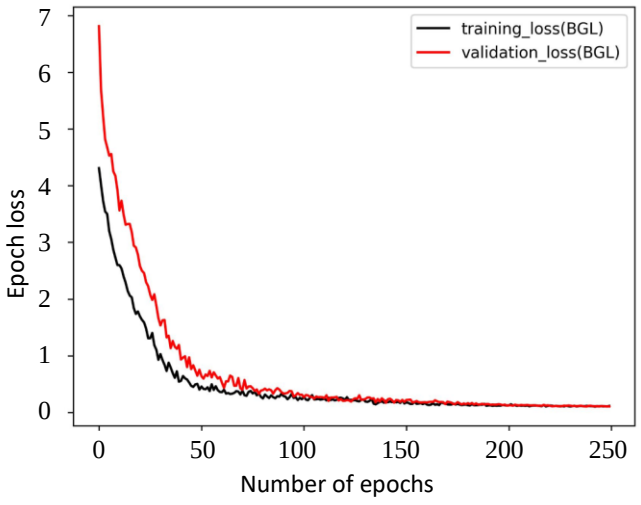
Fig. 4. Few-shot performance of SaRLog. Trained on the BGL dataset and tested on the Thunderbird dataset (orange). Trained on Thunderbird and tested on BGL (blue)

150 epochs to reach a similar state of convergence, as shown in Fig. 2 (a). This result validates the expected decrease in computational complexity and training time associated with utilizing similarity-based learning approaches.

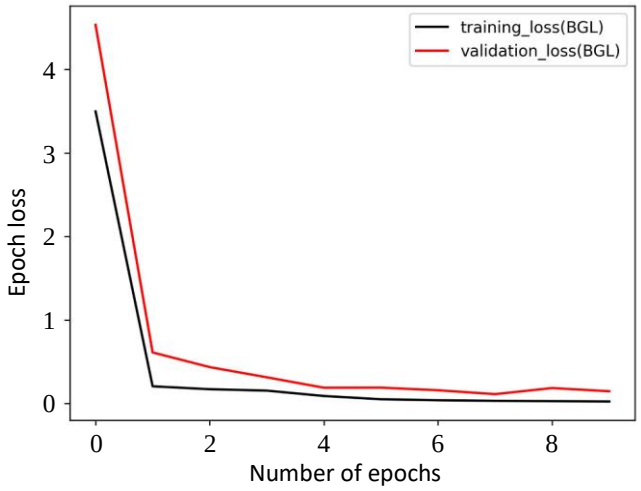
E. Performance of the Representation Model

BERT, renowned for its capacity to grasp contextualized representations and nuanced features, affords a profound comprehension of the input data (Section III). Nonetheless, its generated embeddings may not always exhibit the distinct separability requisite for anomaly detection in specific contexts. This characteristic of BERT embeddings can be attributed to the model's sophisticated architecture and its method of learning contextualized representations, which could lead to overlapping embedding spaces, as illustrated in Fig. 3 (a). In contrast, the custom Siamese network, specifically tailored to discern similarity relationships, produces a more discernible embedding structure when applied in conjunction with BERT's output, as demonstrated in Fig. 3(b). Consequently, this approach simplifies the classification process through the application of straightforward distance metrics, such as cosine or Euclidean distance.

Additionally, empirical analyses were conducted to examine the impact of different representation architectures on the effectiveness of the downstream detection task. Table IV presents the outcomes of model performance employing different representation architectures. The findings highlight the promising performance of domain-specific PLMs, underscoring their efficacy in enhancing the accuracy and efficiency in anomaly detection tasks.



(a)



(b)

Fig. 2. Loss curve trajectory. When using a fully connected neural network-based head and trained for 250 epochs (a). When using Siamese network head and trained only for 10 epochs (b).

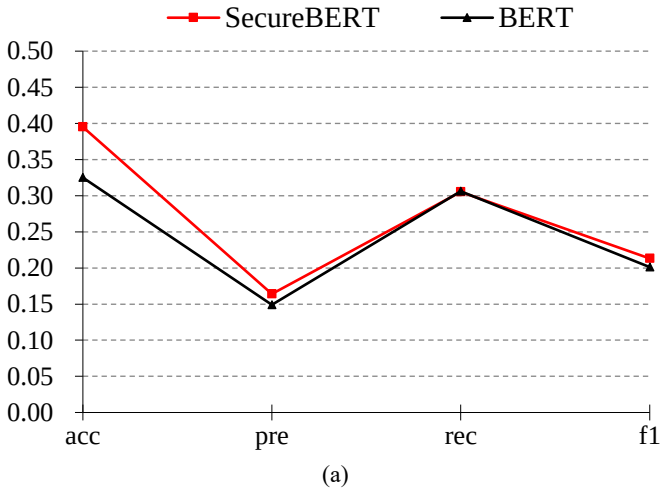
Among the examined embedding architectures, Word2vec demonstrated the lowest performance, yielding an F1-score of 0.885 on BGL and 0.873 on Thunderbird. This outcome suggests its comparatively limited effectiveness compared with more advanced models. GPT-2 exhibits a noteworthy improvement in performance, particularly on the Thunderbird dataset, achieving an F1-score of 0.971. BERT delivered strong results on BGL, attaining an F1-score of 0.963.

Conversely, RoBERTa and SecureBERT emerge as standout performers, particularly on the Thunderbird dataset, where they both achieved nearly perfect F1-score of 0.999. This indicates that these embeddings offer a more nuanced representation, substantially enhancing the model's capacity for generalization and accurate predictions across diverse datasets. On the BGL dataset, RoBERTa and SecureBERT also exhibit robust performance, with F1-scores of 0.973 and 0.988, respectively, further affirming their superiority within the proposed architecture. Overall, SecureBERT consistently outperforms other architectures on both datasets, positioning it as a potentially effective embedding architecture among those considered in the evaluation.

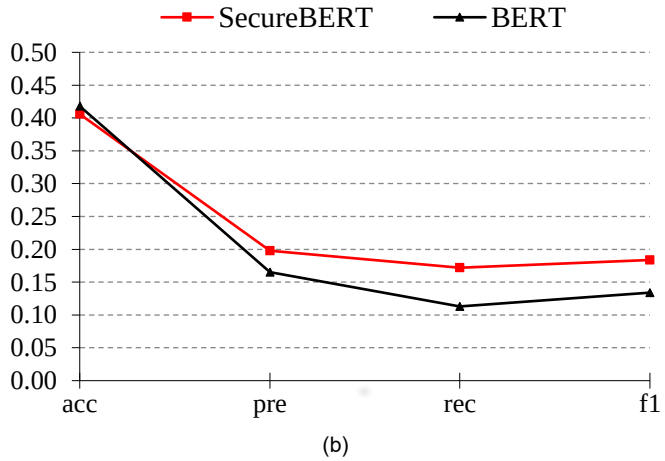
E. Few-Shot Learning Performance of SaRLog

This section discusses the few-shot performance of the model in two distinct scenarios. The evaluation entails training the model on one dataset and subsequently conducting testing on an entirely new dataset, following retraining the model with only a few examples from the new dataset. Fig. 4 shows the performance outcomes, where "TB-BGL" denotes training on Thunderbird and testing on BGL, while "BGL-TB" signifies the reverse scenario.

In the first scenario, where the model initially undergoes training on the BGL dataset, followed by retraining on a subset of examples from the Thunderbird-training dataset, and ultimately testing on the Thunderbird-testing dataset, it achieves an F1 score of 0.871. Conversely, in the second scenario, where the model is exclusively trained and tested on the Thunderbird dataset, followed by retraining with a subset of examples from the BGL dataset and testing on the BGL-test dataset, it achieves a slightly higher F1-score of 0.933. These results highlight the model's capacity to generalize and



(a)



(b)

Fig.5. Zero-shot performance of SaRLog. Trained on the Thunderbird dataset and tested on the BGL dataset (a) and Trained on BGL and tested on Thunderbird (b).

perform effectively with a limited number of examples from a different domain, highlighting its adaptability in few-shot learning scenarios.

VI. Discussion

The proposed model, SaRLog, demonstrates superior performance when compared with baseline methods in both full-training and few-shot learning scenarios. However, when subjected to the zero-shot learning setting, where the model is trained on one dataset and tested on an entirely different one, a significant drop in performance is observed, as illustrated in Fig. 5. This decline can be attributed to structural and distributional variations among log datasets (discussed in Section III), which introduce ambiguity during the representation phase. Notably, the decrease in F1-score from 0.988 to 0.184 when trained on BGL and tested on Thunderbird (Fig. 5b), and vice versa from 0.999 to 0.134 (Fig. 5a), highlights the substantial impact of environment drift on model performance. However, the incorporation of domain-specific PLMs such as SecureBERT yields a slight improvement in performance in both cases, emphasizing the advantages of leveraging domain-specific representations.

To address these challenges fully, future research should prioritize customizing the tokenization process to accommodate domain-specific vocabulary and exploring subtle features that may be common across datasets. Additionally, robust domain adaptation methods should be explored to enhance the model's capacity to generalize to unseen data distributions. These endeavors will contribute to the development of more resilient anomaly detection models capable of effectively handling variations in environment and dataset characteristics.

VII. Conclusions

This study addresses significant challenges in anomaly detection within system logs, with a particular emphasis on striking the delicate balance between model complexity and semantic representation efficacy, while reducing the reliance on extensive labeled data. Despite individual efforts to tackle these challenges, a comprehensive solution has remained elusive. The proposed method, SaRLog, bridges this gap by leveraging the semantic information extraction capabilities of BERT and the few-shot learning ability of a Siamese network.

Through rigorous experimental analysis conducted on the BGL and Thunderbird datasets, SaRLog established its superiority when compared with state-of-the-art methods. It achieved remarkable F1-scores of 0.9880 and 0.9993, respectively, on these datasets. The few-shot learning capability of SaRLog highlights its adaptability and robustness, positioning it as a promising approach for anomaly detection in dynamic and evolving environments. Future research endeavors will be directed toward devising strategies to mitigate the impact of structural and distributional variations in log datasets and exploring robust domain adaptation methods to enhance model generalization, particularly in the zero-shot learning paradigm. These endeavors will contribute to the advancement of anomaly detection solutions that effectively address the challenges posed by diverse and dynamic characteristics of log data.

References

[1] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for Internet of Things: A survey," IEEE Internet of Things J., vol. 3, no. 1, pp. 70–95, Feb. 2016.

[2] A. A. Cook, G. Mısırlı, and Z. Fan, "Anomaly detection for IoT time-series data: A survey," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6481–6494, Jul. 2020.

[3] J. Lou, Q. Fu, S. Yang, Y Xu and J. Li, "Mining invariants from console logs for system problem detection," *ATC'10: Proc. of the USENIX Annual Technical Conference*, 2010.

[4] J. Hojlo, "Future of Industry Ecosystems: Shared Data and Insights," International Data Corporation, Needham, Massachusetts, United States, Jan. 2021. [Online]. Available: https://blogs.idc.com/2021/01/06/future-of-industry-ecosystems-shared-data-and-insights.

[5] A. Oliner and J. Stearley, "What supercomputers say: A study of five system logs," *Proc. 37th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw.* (DSN), pp. 575–584, 2007.

[6] S. He, J. Zhu, P. He and M. R. Lyu, "Loghub: A large collection of system log datasets towards automated log analytics," arXiv:2008.06448, 2020.

[7] S. Lu, X. Wei, Y. Li and L. Wang, "Detecting anomaly in big data system logs using convolutional neural network," *Proc. IEEE 16th Int. Conf. Dependable Auton. Secur. Comput. 16th Int. Conf. Pervasive Intell. Comput. 4th Int. Conf. Big Data Intell. Computing and Cyber Science and Technology Congress*, pp. 159–165, Aug. 2018.

[8] W. Meng et al., "LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs," *Proc. 28th Int. Joint Conf. Artif. Intell.* (IJCAI), vol. 7, pp. 4739–4745, 2019.

[9] Z. Wang, J. Tian, H. Fang, L. Chen and J. Qin, "LightLog: A lightweight temporal convolutional network for log anomaly detection on the edge," *Comput. Netw.*, vol. 203, Feb. 2022.

[10] M. Du, F. Li, G. Zheng and V. Srikumar, "DeepLog: Anomaly detection and diagnosis from system logs through deep learning," *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, pp. 1285–1298, 2017.

[11] X. Zhang et al., "Robust log-based anomaly detection on unstable log data," *Proc. 27th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Foundations Softw. Eng.*, pp. 807–817, 2019.

[12] V.-H. Le and H. Zhang, "Log-based anomaly detection without log parsing," *Proc. IEEE/ACM 36th Int. Conf. Automated Softw. Eng.*, pp. 492–504, 2021.

[13] Y. Lee, J. Kim and P. Kang, "LanoBERT: System log anomaly detection based on BERT masked language model," arXiv:2111.09564, 2021.

[14] L. Yang et al., "Semi-supervised log-based anomaly detection via probabilistic label estimation," in Proc. IEEE/ACM 43rd Int. Conf. Softw. Eng. Companion Proc. (ICSE-Companion). IEEE, 2021, pp. 230–231.

[15] E. Aghaei, X. Niu, W. Shadid and E. Al-Shaer, "SecureBERT: A domain-specific language model for cybersecurity," *Intl. Conf. on Security and Privacy in Communication Systems*, pp. 39–56, 2022.

[16] P. Ranade, A. Piplai, A. Joshi and T. Finin, "CyBERT: Contextualized Embeddings for the Cybersecurity Domain," *2021 IEEE Int. Conf. Big Data (Big Data)*, pp. 3334–3342, December 2021.

[17] L. Yan, C. Luo, and R. Shao, "Discrete log anomaly detection: a novel time-aware graph-based link prediction approach," *Inf. Sci*. 647 (2023): 119576.

[18] A. Makanju, A. N. Zincir-Heywood and E. E. Milios, "Clustering event logs using iterative partitioning," *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, pp. 1255–1264, 2009.

[19] R. Vaarandi, "A data clustering algorithm for mining patterns from Event Logs," *Proc. IEEE Workshop IP Operations and Management*, pp. 119-126, 2003.

[20] P. He, J. Zhu, Z. Zheng and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," *Proc. IEEE Int. Conf. Web Services* (ICWS), pp. 33–40, 2017.

[21] M. Du and F. Li, "Spell: Streaming parsing of system event logs," *Proc. IEEE 16th Int. Conf. Data Mining*, pp. 859–864, 2016.
[22] M. Landauer, F. Skopik, M. Wurzenberger and A. Rauber, "System log clustering approaches for cyber security applications: A survey," *Comput. Security*, vol. 92, May 2020.
[23] Y. Xie and K. Yang, "Domain adaptive log anomaly prediction for hadoop system," *IEEE Internet Things J.*, vol. 9, no. 20, pp. 20778–20787, 15 Oct.15, 2022.
[24] K. Tirumala, A. Markosyan, L. Zettlemoyer and A. Aghajanyan, "Memorization without overfitting: Analyzing the training dynamics of large language models," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 38274–38290, 2022.
[25] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, pp. 4171–4186, 2019.
[26] S. González-Carvajal and E. C. Garrido-Merchán, "Comparing bert against traditional machine learning text classification," arXiv:2005.13012. 2020.
[27] C. Liang et al., "BOND: BERT-assisted open-domain named entity recognition with distant supervision," *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, pp. 1054–1064, 2020.
[28] H. Xu, B. Liu, L. Shu and P. S. Yu, "Bert post-training for review reading comprehension and aspect-based sentiment analysis," arXiv:1904.02232, 2019.
[29] C. Almodovar, F. Sabrina, S. Karimi, and S. Azad, "Can language models help in system security? investigating log anomaly detection using BERT," in *Proc. The 20th Annual Workshop of the Australasian Language Technology Association, Adelaide, Australia*: Australasian Language Technology Association, Dec. 2022, pp. 139–147.
[30] S. Chen and H. Liao, "BERT-log: Anomaly detection for system logs based on pre-trained language model," *Appl. Artif. Intell.*, vol. 36, no. 1, pp. e2145642-1–e2145642-23, Dec. 2022.
[31] Y. Shibata, T. Kida, S. Fukamachi, M. Takeda, A. Shinohara, T. Shinohara, et al., "Byte pair encoding: A text compression scheme that accelerates pattern matching. Technical report," Technical Report DOI-TR-161 Department of Informatics Kyushu University, 1999.

Jilcha Lelisa Adeba received his B.S. degree in Electrical and Computer Engineering from Arbaminch University, Ethiopia in 2015. He has been working as a cybersecurity researcher and supervisor at Information Network Security Administration (INSA), Ethiopia, from 2015 to 2021. He is currently pursuing the M.Sc/PhD program in AI Convergence Network at Ajou University, South Korea. His research interests include deep learning, large language models, intrusion detection, cloud security, convergence security.

K. Deuk-Hun received his B.S degree in information security at Soonchunhyang University in August 2013, with a master's degree in information security at Soonchunhyang University in August 2015 and with a doctor degree in computer engineering at Ajou University in August 2021. He is currently pursuing a post-Doc program in the institute for Computing and Informatics Research, Ajou University. And He is interested in application service security, cloud computing security, cryptography protocol.

J. Kwak received his B.S., M.S., and Ph.D. degrees in Computer Science and Engineering from Sungkyunkwan University, Seoul, Korea, in 2000, 2003, and 2006, respectively. He was Deputy Director of the Ministry of Information and Communication, Korea. Also, he was a Professor of the Department of Information Security Engineering, Soonchunhyang University, Korea. He is currently a Professor of the Department of Cyber Security, Ajou University, Korea. His current research interests include cryptographic protocols, cloud security, SOAR, and applied security services.