A Comparative Study of Fine-Tuned Pretrained BERT Models for Host-Based Intrusion Detection Using the ADFA-LD Dataset

Salsa Zufar Radinka Akmal[1], Tirana Noor Fatyanosa[1], and Mahendra Data[1⋆]

Department of Informatics Engineering, Faculty of Computer Science, Universitas Brawijaya, Malang 65145, Indonesia
salsazufar@gmail.com, fatyanosa@ub.ac.id, mahendra.data@ub.ac.id

Abstract. Host-based intrusion detection uses host activity, including system-call sequences, to distinguish intrusive behavior from normal execution. Although pretrained Bidirectional Encoder Representations from Transformers (BERT) models offer several architectural and computational profiles, their relative behavior for this task has received limited comparative evaluation. This study evaluates five pretrained BERT variants on the Australian Defence Force Academy Linux Dataset (ADFA-LD) under a common binary classification setting. The experiments examine sliding-window n-gram representation and overlap removal, compare fine-tuned model variants, and explore DistilBERT training configurations. In the architecture comparison, BERT-BASE records the highest F1-score of 95.52%, while DistilBERT records 94.98%. DistilBERT also provides the most favorable observed performance-efficiency balance. Its timed evaluation phase, which covered data and model loading and the batched prediction loop, took 44.97 seconds. During this evaluation routine, peak allocated GPU memory was 274.94 MB. Relative to BERT-BASE, these values represent reductions of 60.7% and 37.1%, respectively. The results characterize the practical trade-off between detection performance and computational requirements when selecting a pretrained BERT variant for system-call-based intrusion detection.

Keywords: Host-Based Intrusion Detection · BERT · System Calls · Model Efficiency

1 Introduction

The continuing development of computing systems has expanded both their capabilities and their exposure to cyberattacks. A host-based intrusion detection system (HIDS) examines activity within an individual host, including system-call sequences, and therefore complements evidence available to network-based intrusion detection systems [11]. System calls provide a sequential record of

⋆ Corresponding author.

© The Author(s) 2026

M. Murtadha et al. (eds.), *Proceedings of the 2026 11th International Conference on Sustainable Information Engineering and Technology (SIET 2026)*, Atlantis Highlights in Computer Sciences 26,
https://doi.org/10.2991/978-94-6239-828-3_13

interactions between processes and the operating-system kernel, making them useful for recognizing behavioral deviations associated with an intrusion.

A central challenge for a system-call-based HIDS is distinguishing normal activity from intrusive behavior when their local execution patterns overlap. The Australian Defence Force Academy Linux Dataset (ADFA-LD) is a widely used benchmark containing normal execution and diverse attack scenarios captured on a Linux system. The dataset represents normal and attack executions as ordered system-call traces [3,6]. This structure makes ADFA-LD a relevant setting for studying sequence representations and classification models, while conclusions drawn from it remain specific to its hosts, attacks, and data-collection protocol.

Prior studies on ADFA-LD have considered classical classifiers, including Support Vector Machines and k-Nearest Neighbors, as well as deep sequence models such as Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and hybrid CNN-LSTM architectures [10,9]. These approaches differ in their feature representations, training procedures, and computational requirements. Their reported results motivate further study of models that can learn contextual representations from system-call subsequences, but differences in experimental protocol prevent treating cross-study metrics as direct model rankings.

Transformer-based models have more recently been applied to log and intrusion detection tasks. Chen et al. [1] applied a pretrained Bidirectional Encoder Representations from Transformers (BERT) encoder to parsed event sequences from the Hadoop Distributed File System (HDFS) and Blue Gene/L (BGL) datasets, motivating investigation of pretrained representations for system-call classification. However, numeric syscall identifiers differ substantially from the natural-language data used during pretraining. Existing work provides limited evidence about how BERT-BASE, BERT-LARGE, DistilBERT, ALBERT, and ELECTRA compare under a shared preprocessing and evaluation setting, particularly when computational cost is considered alongside detection metrics.

This study therefore presents a reported within-pipeline comparison of five pretrained BERT variants fine-tuned for binary classification on ADFA-LD. Its objective is to examine how preprocessing choices and model architecture relate to detection performance and measured computational requirements, and to characterize the sensitivity of the selected efficient model to training configurations. The study makes three contributions. First, it evaluates sliding-window n-gram representation and overlap removal for preparing system-call traces. Second, it compares BERT-BASE, BERT-LARGE, DistilBERT, ALBERT, and ELECTRA within the reported experimental pipeline using precision, recall, F1-score, elapsed evaluation time, and peak allocated GPU memory. Third, it analyzes the observed performance-efficiency trade-offs and explores DistilBERT configurations to inform model selection for resource-constrained HIDS settings. These contributions provide an empirical account of architectural and configuration choices within the examined dataset and binary classification task.

2 Related Work

ADFA-LD studies span engineered n-gram and bag-of-words features, classical classifiers, sequence models, and ensembles [6,10]. Ring et al. aggregate recurrent and WaveNet predictions at the application level, whereas Osamor and Wellman use a CNN-LSTM classifier for binary trace classification [11,9]. These studies motivate sequence-aware detection but do not provide baselines under the present n-gram protocol.

Transformer log methods include BERT-Log and LogBERT [1,5], plus NeuralLog and Loader [8,14]. Their units, objectives, and datasets differ from ADFA-LD system-call classification. DistilBERT, ALBERT, and ELECTRA also target different forms of efficiency, so their relative behavior must be measured in the target task and environment [12,7,2].

Reported ADFA-LD outcomes use different splits, labels, preprocessing, evaluation units, aggregation rules, and metrics. The present study therefore treats prior values as context and compares five pretrained BERT variants within the stated ADFA-LD pipeline.

3 Methodology

3.1 Dataset and Preprocessing

The Australian Defence Force Academy Linux Dataset (ADFA-LD) [3] contains Linux system-call traces for host-based intrusion detection. The data used in this study comprise 833 normal traces with 308,077 system calls and 746 attack traces with 317,388 system calls. Normal and attack samples receive binary labels 0 and 1, respectively. The six represented attack families are Adduser, Hydra-FTP, Hydra-SSH, Java Meterpreter, Meterpreter, and Web Shell.

The reported workflow and checked-in pipeline specify that source traces are divided into training, validation, and test sets in a 70/15/15 ratio before subsequence generation. Per-run split artifacts were not recovered, so this ratio cannot be independently confirmed for every result table. The attack traces are stratified by attack family when the corresponding configuration is enabled. Each trace is then processed independently into overlapping, stride-one n-grams, so no n-gram crosses a trace boundary. The reported preprocessing study selected $n = 18$.

Cross-class overlap is handled separately within each split. For a given split, the implementation forms the set intersection of normal and attack n-grams and removes every occurrence of an intersecting n-gram from both classes. This operation does not compare examples across splits and does not remove repeated n-grams that occur only within the same class.

3.2 Input Representation and Models

Each n-gram is serialized as space-separated decimal system-call identifiers and passed to the tokenizer supplied with its pretrained model. This representation

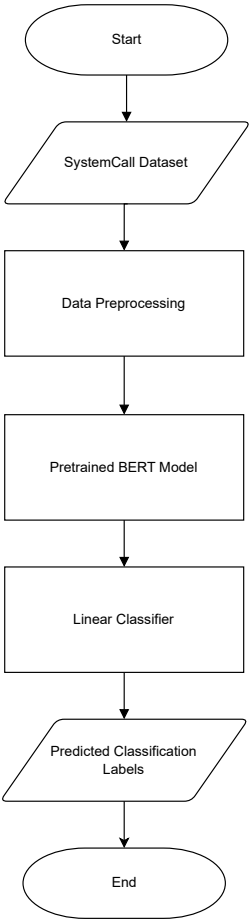


Fig. 1. BERT-based system-call classification pipeline.

enables a common text-classification interface without learning a separate syscall vocabulary. It also creates a domain mismatch because the tokenizer and encoder were pretrained on natural language rather than numeric syscall sequences. The checked-in reference implementation pads or truncates tokenized inputs to 20 tokens; this value is an implementation default, not recovered provenance for every reported run.

The evaluated identifiers follow the published model checkpoints [4,12,7,2]: The evaluated checkpoints were the published BERT-BASE, BERT-LARGE, DistilBERT, ALBERT, and ELECTRA identifiers. Each encoder is fine-tuned with a two-class sequence-classification head. Figure 1 summarizes the pipeline.

3.3 Training Configuration and Metrics

The architecture comparison and the later DistilBERT grid use the same dataset and checked-in preprocessing pipeline, but the retained evidence does not establish identical training and evaluation settings across the two phases. The architecture table is presented in the reported workflow as single-run test-set point estimates. Per-run artifacts were not recovered to independently verify identical test-set provenance across every row, so current code defaults are not assigned retrospectively to those results. The separate DistilBERT grid evaluates batch sizes $\{16, 32\}$, learning rates $\{5 \times 10^{-5}, 3 \times 10^{-5}, 2 \times 10^{-5}\}$, values $\{0.1, 0.15\}$ for the parameter named "warmup ratio," and $\{4, 8\}$ epochs. The grid reports validation F1 for configuration evaluation and selection before any separately reported test evaluation. In the checked-in training code, the "warmup ratio" is the `start_factor` of a `LinearLR` scheduler: the learning rate starts at the selected fraction of its base value and increases linearly toward the base value across the configured training steps. It is therefore not a conventional warmup fraction followed by a separate decay schedule.

The attack class is the positive class. With true positives (TP), false positives (FP), and false negatives (FN), the reported binary metrics are

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \qquad R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \qquad F1 = \frac{2PR}{P + R}. \tag{1}$$

These are positive-class precision, recall, and F1, rather than macro-averaged values.

3.4 Environment and Efficiency Measurement

Experiments were conducted on Ubuntu 24.04.3 LTS with an Intel Xeon Gold 6230R central processing unit (CPU) at 2.10 GHz, 62.8 GB random-access memory (RAM), and an NVIDIA Quadro RTX 8000 graphics processing unit (GPU) with 48 GB memory.

The reported evaluation time is not pure forward-pass latency. The timer starts before peak-memory reset and synchronization and before data loading. It includes reading the test comma-separated values (CSV) file, loading the tokenizer and model, device transfer, dataset and loader construction, tokenization, and the batched prediction loop, then stops immediately after that loop. It excludes the subsequent CUDA synchronization, metric and receiver operating characteristic (ROC) computation, and result serialization. Peak allocated GPU memory is reset before model loading and sampled after the post-loop synchronization. Although the current evaluation default is batch size 16, retained artifacts do not establish the batch size for every reported table row. Reproducibility resources are available in the code repository at `https://github.com/salsazufar/hids_adfa_bert` and the released DistilBERT checkpoint at `https://huggingface.co/salsazufar/distilbert-base-hids-adfa`.

4 Experimental Results

4.1 Evaluation of Data Preprocessing

The reported results for the preprocessing configurations are summarized in Table 1. Without either operation, recall is 99.0% and precision is 69.8%, producing an F1-score of 81.9%. The difference between recall and precision indicates that this configuration produces many false-positive predictions relative to the other reported configurations.

With the sliding-window 6-gram representation, the reported precision and recall are 89.1% and 89.8%, respectively, and F1 is 89.4%. With cross-class overlap removal alone, precision is 76.6%, recall is 96.7%, and F1 is 85.5%. Here, overlap removal denotes the within-split operation defined in the Methodology: sequences present in both the normal and attack classes are removed from both classes. It is not general removal of repeated samples within a class.

Among the four reported configurations, applying sliding-window n-gram segmentation followed by cross-class overlap removal gives the highest observed F1, at 93.0%, with precision of 93.8% and recall of 92.3%. These point estimates show an association between the combined preprocessing configuration and the highest measured values in this comparison, but they do not establish a causal effect.

Table 1. Reported point estimates for the preprocessing configurations. H denotes within-split cross-class overlap removal.

Preprocessing Method	Precision (%)	Recall (%)	F1-Score (%)
No preprocessing	69.8	99.0	81.9
Sliding-window n-gram (S)	89.1	89.8	89.4
Cross-class overlap removal (H)	76.6	96.7	85.5
(S) → (H)	**93.8**	**92.3**	**93.0**

Across the tested n-gram lengths, F1 increases from $n = 4$ to the highest observed value at $n = 18$. At the tested values above 18, precision remains high while recall and F1 vary rather than changing monotonically. The study therefore selects $n = 18$ as the best-observed configuration within this comparison, without claiming that it is optimal beyond the tested values.

4.2 Fine-Tuning of Pretrained Models

Table 2 presents the reported single-run point estimates for the five fine-tuned models. BERT-BASE has the highest observed F1 at 95.52%, followed by DistilBERT at 94.98%, a difference of 0.54 percentage points. The observed difference is small relative to the measured efficiency differences discussed below.

In this run, BERT-LARGE has recall of 100.00% and precision of 58.31%. This combination indicates false-positive-heavy prediction behavior: all observed positive samples are detected, but many samples predicted as attacks are not positive. The available results do not identify whether model capacity, training dynamics, or another factor produced this behavior. ELECTRA and ALBERT have observed F1 values of 94.41% and 93.68%, respectively.

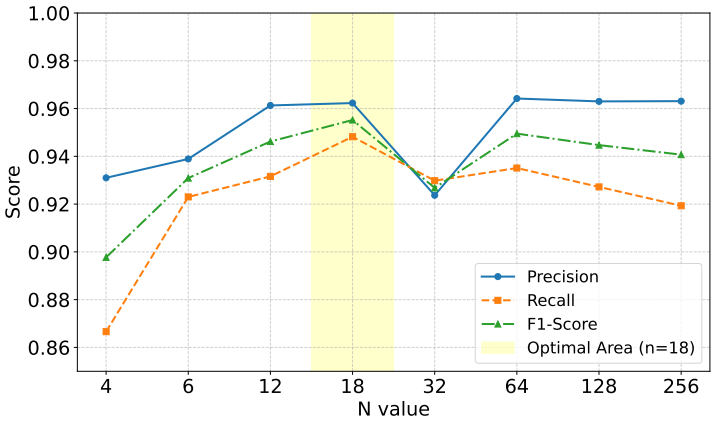


Fig. 2. Performance across tested n-gram lengths; $n = 18$ marks the highest observed F1.

The timed evaluation phase takes 44.97 seconds for DistilBERT and 114.45 seconds for BERT-BASE. Relative to BERT-BASE, the reported DistilBERT value is 60.7% lower, calculated as $(114.45 - 44.97)/114.45$. Its peak allocated GPU memory is 274.94 MB, compared with 437.17 MB for BERT-BASE, a 37.1% reduction calculated as $(437.17 - 274.94)/437.17$. These measurements cover the evaluation routine defined in the Methodology and are not pure model-inference latency. ALBERT has the lowest observed peak memory, at 70.37 MB, whereas BERT-LARGE has both the longest evaluation time, at 193.31 seconds, and the highest peak memory, at 1301.88 MB.

Table 2. Reported single-run performance point estimates and evaluation resource measurements. Evaluation time follows the measurement boundary defined in the Methodology.

Model	Precision (%)	Recall (%)	F1-Score (%)	Evaluation Time (s)	GPU Memory (MB)
BERT-BASE	96.23	94.82	**95.52**	114.45	437.17
BERT-LARGE	58.31	**100.00**	73.66	193.31	1301.88
DistilBERT	**96.42**	93.57	94.98	**44.97**	274.94
ALBERT	94.50	92.87	93.68	89.30	**70.37**
ELECTRA	95.35	93.48	94.41	74.40	437.76

Under the three objectives of maximizing F1 while minimizing evaluation time and peak memory, BERT-BASE, DistilBERT, and ALBERT are nondominated among the five reported points. BERT-BASE defines the highest observed F1, DistilBERT defines the lowest observed evaluation time, and ALBERT defines the lowest observed peak memory. DistilBERT dominates ELECTRA on all three reported dimensions, but it does not dominate BERT-BASE or ALBERT. These results support different choices according to the resource constraint rather than a single universally superior model.

The observed DistilBERT trade-off is consistent with the design objective of knowledge distillation. Sanh et al. [12] describe a compact student model pretrained to approximate a larger BERT teacher, with approximately 40% fewer parameters while retaining much of BERT's reported downstream performance.

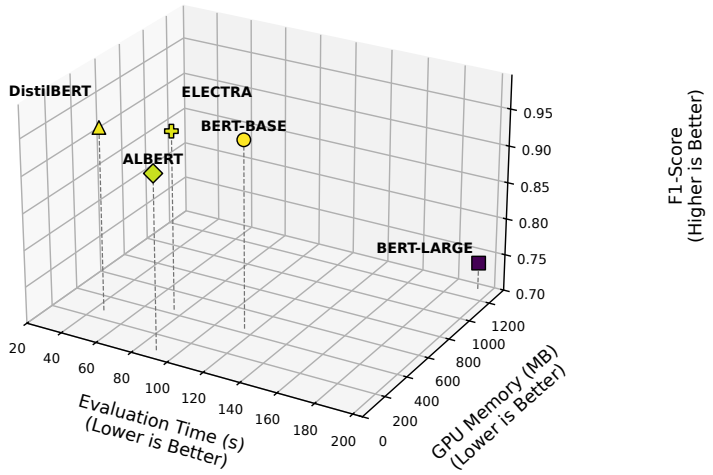


Fig. 3. Observed trade-offs among F1, evaluation time, and peak allocated GPU memory.

In the present comparison, DistilBERT's observed F1 is 0.54 percentage points below BERT-BASE, while its evaluation time and peak memory are lower by 60.7% and 37.1%, respectively. This observed balance motivated its selection for the subsequent configuration exploration.

4.3 DistilBERT Configuration Exploration

After the architecture comparison, a grid exploration was conducted to characterize DistilBERT across batch sizes $\{16, 32\}$, learning rates $\{5 \times 10^{-5}, 3 \times 10^{-5}, 2 \times 10^{-5}\}$, values $\{0.1, 0.15\}$ for the parameter named "warmup ratio," and $\{4, 8\}$ epochs. The reported validation F1 values range from 93.18% to 94.49%, with a mean of 93.87% and a standard deviation of 0.36 percentage points across the heterogeneous configurations. This standard deviation describes variation across configuration choices. It is not an estimate of uncertainty across random seeds or repeated runs.

The explored grid contains both batch sizes, both "warmup ratio" values, and both epoch counts. Because several factors change across configurations and each configuration has one reported value, the grid does not isolate the independent influence of any single hyperparameter.

The highest observed validation F1, 94.49%, occurs at batch size 32, learning rate 2×10^{-5}, "warmup ratio" 0.1, and eight epochs. This value is lower than the 94.98% DistilBERT test-set point estimate in the earlier architecture comparison. Since the two values come from different experimental phases and evaluation sets, the grid result cannot be described as an improvement over the architecture result. It identifies the highest observed validation configuration within the reported grid, not a universally optimal setting.

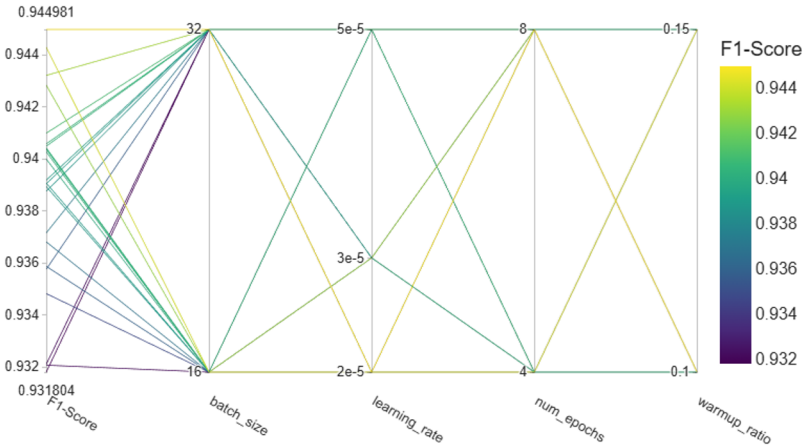


Fig. 4. Explored DistilBERT configurations and reported validation F1.

5 Discussion

5.1 Interpreting the Performance-Efficiency Trade-off

The architecture comparison addresses the study objective by showing that detection performance and measured resource use do not vary in the same direction across the evaluated models. BERT-BASE records the highest observed F1, whereas DistilBERT requires substantially less evaluation time and peak allocated GPU memory with a small observed F1 difference. ALBERT establishes a third trade-off by minimizing peak allocated memory at the expense of lower F1 and longer evaluation time than DistilBERT. These observations support treating model selection as a multi-objective decision rather than equating the largest architecture or the highest point estimate with the most suitable deployment choice. The tested representation configurations were associated with changes in the observed precision-recall profile, while architecture choice was associated with different observed performance-resource trade-offs.

The external studies report engineered-feature and sequence-model results under different evaluation units and splits [10,11], as well as different labels, preprocessing, aggregation rules, and metrics [9,13]. Their values therefore provide context rather than protocol-equivalent baselines or a basis for direct numerical ranking against the present comparison.

5.2 Deployment Implications

The five variants support different choices. BERT-BASE offers the highest observed F1, DistilBERT the best observed balance of F1, time, and memory, and ALBERT the lowest peak memory. ELECTRA is intermediate, while BERT-LARGE combines the highest resource cost with false-positive-heavy predictions. The BERT-BASE versus DistilBERT difference has not been shown to

be statistically distinguishable, and the evidence does not support an overfitting diagnosis for BERT-LARGE.

These offline measurements do not establish production readiness. A live HIDS would additionally require throughput, alert latency, resource-contention, n-gram aggregation, threshold calibration, false-alarm, drift, update, and rollback evaluation, none of which was performed here.

5.3 Limitations

The evidence is bounded to one dataset and a binary normal-versus-attack task, without per-attack evaluation or live-host validation, so it does not establish performance on other hosts, datasets, or attack-specific classification. Because the reported architecture results are based on one reported run per model, the small F1 difference between BERT-BASE and DistilBERT should not be interpreted as evidence of a statistically significant performance difference. Repeated-seed evaluation remains necessary to quantify this uncertainty. In addition, serializing numeric syscall identifiers for tokenizers pretrained on natural language creates a tokenizer and domain mismatch whose effect was not isolated. Exact historical software and runtime versions and the per-run split, prediction, and timing artifacts were not recovered, which limits independent reconstruction of the reported measurements.

6 Conclusions and Future Work

This study compared representations, five pretrained BERT variants, and DistilBERT configurations for binary HIDS on ADFA-LD. N-gram construction and within-split cross-class overlap removal were associated with changes in precision, recall, and F1; $n = 18$ produced the highest observed F1.

BERT-BASE produced the highest observed single-run F1 at 95.52%. DistilBERT reached 94.98% F1 with 60.7% lower evaluation time and 37.1% lower peak GPU memory than BERT-BASE. ALBERT used the least memory, ELECTRA was intermediate, and BERT-LARGE was false-positive-heavy and most costly. The DistilBERT grid used validation F1 and yielded 94.49% at best, with mean 93.87% and standard deviation 0.36 across configurations.

Future work should evaluate repeated random seeds and additional HIDS datasets, report performance by attack category, and validate the pipeline on live hosts or streaming system-call data. Operational evaluation should also define how n-gram predictions are aggregated into alerts and assess score calibration, throughput, latency, drift, and false-alarm handling.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Chen, S., Liao, H.: BERT-Log: Anomaly detection for system logs based on pre-trained language model. Applied Artificial Intelligence **36**(1), 2145642 (Dec 2022). https://doi.org/10.1080/08839514.2022.2145642
2. Clark, K., Luong, M.T., Le, Q.V., Manning, C.D.: ELECTRA: Pre-training text encoders as discriminators rather than generators (Mar 2020). https://doi.org/10.48550/arXiv.2003.10555
3. Creech, G., Hu, J.: Generation of a new IDS test dataset: Time to retire the KDD collection. In: 2013 IEEE Wireless Communications and Networking Conference (WCNC). pp. 4487–4492. IEEE (Apr 2013). https://doi.org/10.1109/WCNC.2013.6555301
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding (May 2019). https://doi.org/10.48550/arXiv.1810.04805
5. Guo, H., Yuan, S., Wu, X.: LogBERT: Log anomaly detection via BERT. In: 2021 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2021). https://doi.org/10.1109/IJCNN52387.2021.9534113
6. Khandelwal, P., Likhar, P., Yadav, R.S.: Machine learning methods leveraging ADFA-LD dataset for anomaly detection in linux host systems. In: 2022 2nd International Conference on Intelligent Technologies (CONIT). pp. 1–8. IEEE (Jun 2022). https://doi.org/10.1109/CONIT55038.2022.9848305
7. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: A lite BERT for self-supervised learning of language representations (Feb 2020). https://doi.org/10.48550/arXiv.1909.11942
8. Le, V.H., Zhang, H.: Log-based anomaly detection without log parsing. In: 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE). pp. 492–504. IEEE (2021). https://doi.org/10.1109/ASE51524.2021.9678773
9. Osamor, F., Wellman, B.: Deep learning-based hybrid model for efficient anomaly detection. International Journal of Advanced Computer Science and Applications **13**(4) (2022). https://doi.org/10.14569/IJACSA.2022.01304111
10. Patel, A.: Enhancing intrusion detection systems through advanced feature engineering and machine learning techniques. In: 2024 2nd International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT). vol. 1, pp. 941–948. IEEE (Nov 2024). https://doi.org/10.1109/ICAICCIT64383.2024.10912389
11. Ring, J.H., Van Oort, C.M., Durst, S., White, V., Near, J.P., Skalka, C.: Methods for host-based intrusion detection with deep learning. Digital Threats: Research and Practice **2**(4), 1–29 (Dec 2021). https://doi.org/10.1145/3461462
12. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter (Mar 2020). https://doi.org/10.48550/arXiv.1910.01108
13. Satilmiş, H., Akleylek, S., Yüce Tok, Z.: Development of various stacking ensemble-based HIDS using ADFA datasets. IEEE Open Journal of the Communications Society **6**, 1170–1189 (2025). https://doi.org/10.1109/OJCOMS.2025.3538101
14. Xiao, T., Quan, Z., Wang, Z.J., Le, Y., Du, Y., Liao, X., Li, K., Li, K.: Loader: A log anomaly detector based on transformer. IEEE Transactions on Services Computing **16**(5), 3479–3492 (2023). https://doi.org/10.1109/TSC.2023.3280575

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (http://creativecommons.org/licenses/by-nc/4.0/), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

