User Behaviour Analysis for Insider Threat Detection Using Machine Learning: A Case Study in Enterprise Web Application Security

Yosep, Aditya Kurniawan
Department of Computer Science, Binus University, Jakarta, Indonesia

***Abstract*—This study presents a user behaviour analysis approach for detecting insider threats in an enterprise web application environment. The approach applies machine learning techniques to analyze patterns of user activity. Using a primary dataset collected from a leading ICT distributor company in Indonesia with nationwide channel operations over January–June 2025, we identify patterns of normal and anomalous user activities indicative of insider threats. Three machine learning models were implemented: Random Forest, Support Vector Machine (SVM) with RBF kernel, and 1D CNN, which are widely used in insider-threat and anomaly-detection research. Severe class imbalance was mitigated via undersampling followed by SMOTE. Random Forest delivered the best performance on the test set (Accuracy 97.38%, F1-Score 97.77%, ROC-AUC 99.82%), with CNN and SVM also showing strong anomaly sensitivity. The findings demonstrate a practical, high-accuracy insider-threat detector trained on real enterprise logs, not simulated datasets, suitable for deployment in Indonesian enterprise settings.**

***Keywords*—Insider threat; user behaviour analytics; machine learning; anomaly detection; cybersecurity**

I. Introduction

Insider threat refers to security risks originating from individuals who have authorized access to organizational systems but misuse that access, either intentionally or unintentionally [1]. To address this challenge, user behavior analytics has emerged as an important approach for monitoring and analyzing patterns of user activity within enterprise systems [2]. Machine learning techniques have been widely applied in this domain due to their ability to identify complex and hidden behavioral patterns in large-scale log data [3]. However, insider threat detection remains challenging because such datasets are typically highly imbalanced, where anomalous activities occur far less frequently than normal behavior [4]. Recent studies also highlight that behavior log analysis using real enterprise data provides a more practical and realistic foundation for developing insider threat detection systems [5]. Insider threat is one of the most critical challenges in modern enterprise security [6], especially within web-based business applications [7]. Recent reports show that 83% of companies experienced insider-related incidents with an average annual loss of USD 16.8 million [8], while user behavior analytics has become an increasingly important capability in enterprise security monitoring. In many cases, insider activity is difficult to detect because malicious behaviour is often hidden behind legitimate access privileges [9].

One of Indonesia's leading information and communication technology (ICT) distributor companies, with nationwide operational coverage, operates multiple internal web applications used by employees, partners, and operational staff. These applications generate extensive activity logs that capture user identity, device, IP, access time, request methods, and accessed URLs. Traditional monitoring methods, such as manual log inspection or periodic vulnerability scanning, are inadequate to detect subtle behavioural anomalies [10]. General network anomaly detection techniques have also been widely studied in the literature, providing foundational methods for detecting abnormal patterns in large-scale data [20]. In addition, the broader web-application environment is increasingly exposed to evolving security risks and attack surfaces that require more adaptive monitoring approaches [11].

This study addresses the following problem: how to automatically detect suspicious user behaviour indicative of insider threats without disrupting normal user activity?

The research develops a machine-learning-based User Behaviour Analysis (UBA) system using enterprise log data to identify anomalous access patterns [12]. Using Random Forest, SVM, and CNN, the system learns normal behaviour profiles and identifies deviations that may signal insider threats [13].

The contributions of this research are:

- Development of an insider-threat detection approach using real enterprise log data (instead of public CERT datasets) [14].
- Handling of extreme class imbalance using undersampling and SMOTE [15].
- Comparative evaluation of three machine learning models on real operational data [16].
- Practical insights for implementation in Indonesian enterprise environments [17].

II. Related Work

Research on insider threat detection has grown significantly in recent years, with a strong emphasis on behavioural analytics, machine learning, and deep learning approaches [18]. Most studies rely on the CERT Insider Threat Dataset, a synthetic dataset commonly used for benchmarking anomaly-detection models [19].

Key findings from prior studies include:

- Rasheed et al. [15] applied Random Forest, and ANN combined with SMOTE to address class imbalance, achieving accuracy up to 99.8%, demonstrating the effectiveness of oversampling techniques in improving classifier performance.
- Bin Sarhan & Altwaijry [3] compared multiple ML algorithms (Random Forest, Decision Tree, SVM, Logistic Regression, Naïve Bayes) and found that Random Forest achieved the highest accuracy (99.2%), while feature selection significantly improved overall results.
- Kim & Lee [16] proposed an LSTM autoencoder model trained on anonymized user activity logs. Despite de-identification using the ARX tool, the model achieved 94% accuracy and 97% F1-score, showing that anonymized data can still be effective for detecting anomalies.
- Ojo et al. [22] evaluated several ML algorithms using SMOTE and found that Random Forest reached 100% accuracy and 93% recall, confirming its robustness as an anomaly-detection model.

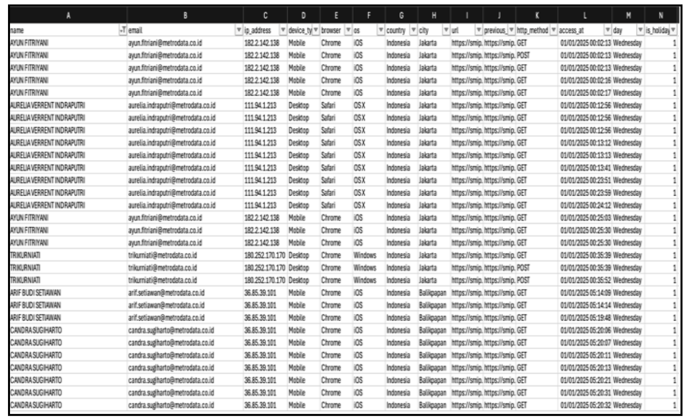
Compared with previous studies, Random Forest-based approaches consistently demonstrate strong performance in handling high-dimensional behavioral datasets. Deep learning models such as LSTM and CNN have also shown promising results in capturing temporal patterns in user activity logs. More recent studies have explored hybrid approaches for insider threat detection [23]. Sequence-based and transformer-based models have also been proposed to capture temporal user behaviour patterns [24]. However, most prior work relies on simulated datasets such as CERT, which may not fully reflect real enterprise environments. Therefore, this study focuses on evaluating multiple machine learning models using real operational logs to provide a more practical assessment of insider-threat detection performance [25].

III. Methods

A. Dataset

The dataset used in this study was derived from user activity logs collected from internal enterprise web applications operated by a leading information and communication technology (ICT) distribution company in Indonesia. The dataset consists of activity records generated by 1455 active internal users interacting with enterprise web applications during daily operational activities. The recorded actions include page navigation, data access requests, and transaction-related operations performed through the web interface. These activity types provide diverse behavioral patterns that are useful for distinguishing normal operational behavior from potentially suspicious access patterns. The data covers a continuous observation period from January to June 2025 and represents real operational usage within a production environment. Each log record is labeled into two behavioral classes, namely normal user activity and suspicious or anomalous activity, which may indicate potential insider-threat behavior. The labeling process was conducted in collaboration with domain experts to ensure that anomalous activities reflected realistic security-related deviations rather than random noise.

The collected logs capture a wide range of contextual and behavioral attributes, including device characteristics, browser type, and operating system information, as well as network-level details such as IP address, country, and city of access. In addition, each record contains request-level information, including the accessed URL, HTTP request method, and precise timestamps of user actions. Temporal features were further enriched by extracting the hour-of-access and incorporating calendar-based indicators, such as weekday and national holiday flags, to capture potential behavioral variations related to working schedules and non-working days. An illustrative example of the dataset structure is presented in Fig. 1.



A	B	C	D	E	F	G	H	I	J	K	L	M	N
name	email	ip_address	device_ty	browser	os	country	city	url	previous_	http_method	access_at	day	is_holiday
AYUN FITRIYANI	ayun.fitriani@metrodata.co.id	182.2.142.138	Mobile	Chrome	iOS	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:02:13	Wednesday	1
AYUN FITRIYANI	ayun.fitriani@metrodata.co.id	182.2.142.138	Mobile	Chrome	iOS	Indonesia	Jakarta	https://smip.	https://smip.	POST	01/01/2025 00:02:13	Wednesday	1
AYUN FITRIYANI	ayun.fitriani@metrodata.co.id	182.2.142.138	Mobile	Chrome	iOS	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:02:13	Wednesday	1
AYUN FITRIYANI	ayun.fitriani@metrodata.co.id	182.2.142.138	Mobile	Chrome	iOS	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:02:16	Wednesday	1
AYUN FITRIYANI	ayun.fitriani@metrodata.co.id	182.2.142.138	Mobile	Chrome	iOS	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:02:17	Wednesday	1
AURELIA VERRENT INDRAPUTRI	aurelia.indraputri@metrodata.co.id	111.94.1.213	Desktop	Safari	OSX	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:12:56	Wednesday	1
AURELIA VERRENT INDRAPUTRI	aurelia.indraputri@metrodata.co.id	111.94.1.213	Desktop	Safari	OSX	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:12:56	Wednesday	1
AURELIA VERRENT INDRAPUTRI	aurelia.indraputri@metrodata.co.id	111.94.1.213	Desktop	Safari	OSX	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:12:56	Wednesday	1
AURELIA VERRENT INDRAPUTRI	aurelia.indraputri@metrodata.co.id	111.94.1.213	Desktop	Safari	OSX	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:13:12	Wednesday	1
AURELIA VERRENT INDRAPUTRI	aurelia.indraputri@metrodata.co.id	111.94.1.213	Desktop	Safari	OSX	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:13:13	Wednesday	1
AURELIA VERRENT INDRAPUTRI	aurelia.indraputri@metrodata.co.id	111.94.1.213	Desktop	Safari	OSX	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:13:41	Wednesday	1
AURELIA VERRENT INDRAPUTRI	aurelia.indraputri@metrodata.co.id	111.94.1.213	Desktop	Safari	OSX	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:23:51	Wednesday	1
AURELIA VERRENT INDRAPUTRI	aurelia.indraputri@metrodata.co.id	111.94.1.213	Desktop	Safari	OSX	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:23:59	Wednesday	1
AURELIA VERRENT INDRAPUTRI	aurelia.indraputri@metrodata.co.id	111.94.1.213	Desktop	Safari	OSX	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:24:12	Wednesday	1
AYUN FITRIYANI	ayun.fitriani@metrodata.co.id	182.2.142.138	Mobile	Chrome	iOS	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:25:03	Wednesday	1
AYUN FITRIYANI	ayun.fitriani@metrodata.co.id	182.2.142.138	Mobile	Chrome	iOS	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:25:30	Wednesday	1
AYUN FITRIYANI	ayun.fitriani@metrodata.co.id	182.2.142.138	Mobile	Chrome	iOS	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:25:30	Wednesday	1
TRIKURNIATI	trikurniati@metrodata.co.id	180.252.170.170	Desktop	Chrome	Windows	Indonesia	Jakarta	https://smip.	https://smip.	GET	01/01/2025 00:35:39	Wednesday	1
TRIKURNIATI	trikurniati@metrodata.co.id	180.252.170.170	Desktop	Chrome	Windows	Indonesia	Jakarta	https://smip.	https://smip.	POST	01/01/2025 00:35:39	Wednesday	1
TRIKURNIATI	trikurniati@metrodata.co.id	180.252.170.170	Desktop	Chrome	Windows	Indonesia	Jakarta	https://smip.	https://smip.	POST	01/01/2025 00:35:52	Wednesday	1
ARIF BUDI SETIAWAN	arif.setiawan@metrodata.co.id	36.85.39.101	Mobile	Chrome	iOS	Indonesia	Balikpapan	https://smip.	https://smip.	GET	01/01/2025 05:14:09	Wednesday	1
ARIF BUDI SETIAWAN	arif.setiawan@metrodata.co.id	36.85.39.101	Mobile	Chrome	iOS	Indonesia	Balikpapan	https://smip.	https://smip.	GET	01/01/2025 05:14:14	Wednesday	1
ARIF BUDI SETIAWAN	arif.setiawan@metrodata.co.id	36.85.39.101	Mobile	Chrome	iOS	Indonesia	Balikpapan	https://smip.	https://smip.	GET	01/01/2025 05:19:48	Wednesday	1
CANDRA SUGIHARTO	candra.sugiharto@metrodata.co.id	36.85.39.101	Mobile	Chrome	iOS	Indonesia	Balikpapan	https://smip.	https://smip.	GET	01/01/2025 05:20:06	Wednesday	1
CANDRA SUGIHARTO	candra.sugiharto@metrodata.co.id	36.85.39.101	Mobile	Chrome	iOS	Indonesia	Balikpapan	https://smip.	https://smip.	GET	01/01/2025 05:20:07	Wednesday	1
CANDRA SUGIHARTO	candra.sugiharto@metrodata.co.id	36.85.39.101	Mobile	Chrome	iOS	Indonesia	Balikpapan	https://smip.	https://smip.	GET	01/01/2025 05:20:11	Wednesday	1
CANDRA SUGIHARTO	candra.sugiharto@metrodata.co.id	36.85.39.101	Mobile	Chrome	iOS	Indonesia	Balikpapan	https://smip.	https://smip.	GET	01/01/2025 05:20:13	Wednesday	1
CANDRA SUGIHARTO	candra.sugiharto@metrodata.co.id	36.85.39.101	Mobile	Chrome	iOS	Indonesia	Balikpapan	https://smip.	https://smip.	GET	01/01/2025 05:20:21	Wednesday	1
CANDRA SUGIHARTO	candra.sugiharto@metrodata.co.id	36.85.39.101	Mobile	Chrome	iOS	Indonesia	Balikpapan	https://smip.	https://smip.	GET	01/01/2025 05:20:31	Wednesday	1
CANDRA SUGIHARTO	candra.sugiharto@metrodata.co.id	36.85.39.101	Mobile	Chrome	iOS	Indonesia	Balikpapan	https://smip.	https://smip.	GET	01/01/2025 05:20:32	Wednesday	1

Fig. 1. Dataset example.

Before model development, a comprehensive data preprocessing phase was conducted to ensure data quality, privacy compliance, and computational efficiency. All sensitive personal identifiers, including usernames and email addresses, were permanently removed to preserve confidentiality and comply with enterprise data governance policies. Timestamp attributes were transformed into numerical representations, enabling machine learning models to effectively capture temporal access patterns. Categorical variables, such as device type, browser, operating system, and geographical location, were converted into numerical form using Label Encoding to maintain compatibility with traditional machine learning algorithms. Furthermore, numerical attributes were downcast to lower-precision data types to reduce memory consumption and improve processing efficiency when handling large-scale log data. To ensure fair and representative evaluation, the dataset was partitioned using a stratified splitting strategy, allocating 80% of the data for training, 10% for validation, and 10% for testing, while preserving the original class distribution across all subsets. The overall distribution of the dataset across classes is illustrated in Fig. 2.

A significant challenge in this dataset is the presence of severe class imbalance, a common characteristic of insider threat detection scenarios, where malicious or anomalous behaviors occur far less frequently than normal user activities. To address this issue and prevent model bias toward the majority class, resampling techniques were applied exclusively to the training set. Specifically, random undersampling was used to reduce the

number of normal activity samples to 150,000 instances, thereby limiting majority class dominance. Subsequently, the Synthetic Minority Over-sampling Technique (SMOTE) was applied to generate synthetic anomalous samples, increasing the minority class proportion to a ratio of 0.40. This combined resampling strategy enabled the models to learn discriminative patterns from both classes more effectively, while preserving realistic behavioral distributions for validation and testing. The resulting balanced training data provides a robust foundation for evaluating machine learning-based user behaviour analysis in the context of enterprise web application security [21].

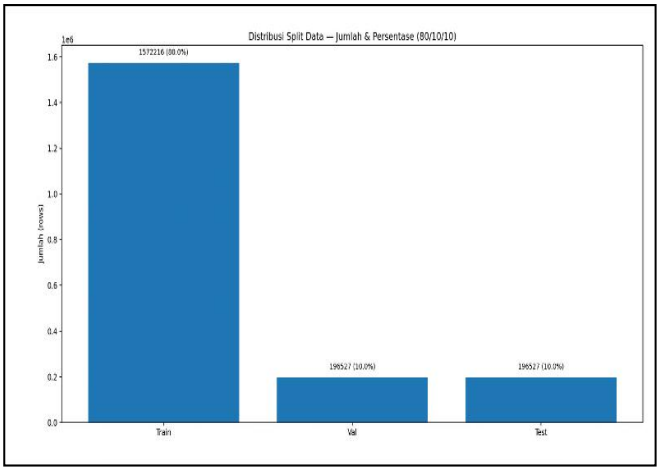


Fig. 2. Dataset distribution.

B. Model Architecture

In this study, three distinct machine learning models were designed, implemented, and evaluated to analyze user behaviour patterns for insider threat detection within an enterprise web application environment. The selected models represent complementary learning paradigms, encompassing ensemble-based methods, margin-based classifiers, and deep learning architectures. This diversified modeling approach enables a comprehensive assessment of both classical and neural network–based techniques when applied to structured log data containing temporal, categorical, and contextual features. By comparing models with different inductive biases, this study aims to identify architectures that are not only accurate but also robust and practical for deployment in real-world enterprise security systems.

The Random Forest classifier was adopted as the baseline model due to its proven effectiveness in handling high-dimensional tabular datasets and its inherent resistance to overfitting. As an ensemble learning method based on multiple decision trees, Random Forest is particularly well-suited for capturing non-linear relationships and complex feature interactions commonly found in user activity logs. Additionally, its ability to provide feature importance scores offers a degree of interpretability that is valuable for security analysts seeking to understand the behavioral indicators contributing to anomaly detection [25]. Hyperparameter tuning was performed using the validation set to balance predictive performance and computational efficiency. The final configuration consisted of 200 decision trees with a maximum depth of 18, allowing the model to learn sufficiently deep behavioral patterns without excessive complexity. Feature selection at each split was controlled using the square-root strategy, while the maximum number of samples per tree was limited to 150,000 to align with the resampled training data and reduce training variance.

To complement the ensemble-based approach, a Support Vector Machine (SVM) classifier with a radial basis function (RBF) kernel was implemented to model complex decision boundaries in high-dimensional feature spaces. SVMs are particularly effective in scenarios where class separation is non-linear, making them suitable for distinguishing subtle behavioral deviations associated with insider threats [26]. The RBF kernel enables the transformation of input features into a higher-dimensional space where linear separation becomes feasible. After empirical tuning, the regularization parameter was set to a value of 1.0 to achieve a balance between margin maximization and classification error, while the kernel coefficient was configured using the scale heuristic to adapt automatically to the variance of the input features. This configuration allows the SVM model to maintain stable generalization performance without excessive sensitivity to noisy or overlapping data points.

In addition to classical machine learning models, a one-dimensional Convolutional Neural Network (1D CNN) was developed to capture sequential and local temporal patterns within the user activity logs [27]. Although originally popularized for signal and text processing, 1D CNNs have demonstrated strong performance in learning structured patterns from ordered tabular data when temporal or sequential relationships are present. The proposed architecture begins with a one-dimensional convolutional layer consisting of 64 filters and a kernel size of three, enabling the model to extract localized behavioral patterns across adjacent feature sequences. To mitigate overfitting and enhance generalization, dropout regularization was applied after the convolutional layer, followed by a fully connected dense layer with 64 neurons and an additional dropout stage. Global average pooling was then used to reduce dimensionality and stabilize training by aggregating learned feature maps before the final classification stage. The network employs a sigmoid activation function at the output layer to produce probabilistic predictions for binary classification. Model training was conducted using the Adam optimizer with binary cross-entropy as the loss function, over 15 training epochs and a batch size of 256, which provided an effective trade-off between convergence speed and model stability.

Since all evaluated models produce probabilistic output scores rather than discrete class labels, a unified threshold optimization strategy was applied to ensure fair comparison and optimal classification performance. Instead of relying on a fixed default threshold, the decision boundary for each model was determined using Youden's J statistic, a widely adopted metric in binary classification tasks. This approach selects the threshold that maximizes the difference between the true positive rate and the false positive rate on the validation set, thereby achieving an optimal balance between detection sensitivity and false alarm reduction. The use of Youden's J is particularly relevant in insider threat detection scenarios, where minimizing false negatives is critical, yet excessive false positives can overwhelm security teams. By applying this thresholding strategy consistently across all models, the evaluation framework

ensures that performance comparisons reflect genuine model capabilities rather than arbitrary decision thresholds [28].

IV. Results and Discussion

A. Experimental Setup

All experiments in this study were conducted within a controlled software and hardware environment to ensure reproducibility and consistency of the reported results. The implementation was carried out using Python version 3.9.6 as the primary programming language, selected for its extensive ecosystem of machine learning and data analysis libraries. The experiments were executed on a MacBook Pro 2020 equipped with 8 GB of main memory, representing a modest yet realistic development environment commonly available to practitioners in enterprise and academic settings. By intentionally avoiding high-performance computing infrastructure, this study aims to demonstrate that effective insider threat detection models can be developed and evaluated using standard workstation-level hardware, thereby enhancing the practical applicability of the proposed approach.

Data processing, model development, and evaluation were supported by a collection of widely adopted open-source libraries. The pandas and NumPy libraries were used for efficient data manipulation, preprocessing, and numerical computation, enabling scalable handling of large volumes of log data. Visualization tasks, including performance curves and result summaries, were performed using Matplotlib to provide clear graphical representations of model behavior. Classical machine learning models, such as Random Forest and Support Vector Machine, were implemented using the scikit-learn framework, which offers reliable implementations and consistent evaluation utilities. To address the class imbalance inherent in insider threat datasets, the imbalanced-learn library was employed to perform both undersampling and oversampling techniques during the training phase. For deep learning experiments, the TensorFlow framework with the Keras high-level API was utilized to design, train, and optimize the one-dimensional convolutional neural network, allowing for efficient experimentation with neural architectures and training configurations.

Model performance was assessed using a comprehensive set of evaluation metrics designed to capture both overall predictive accuracy and class-specific detection capabilities. Accuracy was used as an initial measure of correct classification across all samples; however, given the imbalanced nature of the dataset, additional metrics were emphasized to provide a more nuanced evaluation. Precision and recall were calculated to assess the model's ability to correctly identify anomalous user activities while minimizing false alarms and missed detections, respectively. The F1-score was included as a harmonic mean of precision and recall, offering a balanced metric that reflects both detection reliability and sensitivity. Furthermore, the area under the Receiver Operating Characteristic curve (ROC-AUC) was employed as a threshold-independent metric to evaluate the discriminative power of each model across varying decision thresholds. Together, these metrics provide a robust and comprehensive evaluation framework for assessing machine learning models in the context of enterprise insider-threat detection.

B. Validation Results

The validation performance of the evaluated models is illustrated in Fig. 3–Fig. 5 and summarized in Table I.

TABLE I. Validation Performance Comparison

Model	Accuracy	Precision	Recall	F1 Score
Random Forest	97,39	98,60	97,39	97,78
SVM	92,96	97,73	92,96	94,73
CNN	94,52	97,94	94,52	95,75

```
Training RandomForest...
Best threshold (Youden's J on Validation): 0.2719

[INFO] Using threshold = 0.2719

=== Validation Report ===
              precision    recall  f1-score   support

           0     0.9996    0.9736    0.9864    190969
           1     0.5206    0.9852    0.6813      5558

    accuracy                         0.9739    196527
   macro avg     0.7601    0.9794    0.8338    196527
weighted avg     0.9860    0.9739    0.9778    196527

Confusion Matrix (Val):
[[185927   5042]
 [    82   5476]]
Validation ROC AUC : 0.9983
```

Fig. 3. Random forest performance data validation.

```
[INFO] Using threshold = 0.1651

=== Validation Report ===
              precision    recall  f1-score   support

           0     0.9977    0.9297    0.9625    190969
           1     0.2769    0.9253    0.4263      5558

    accuracy                         0.9296    196527
   macro avg     0.6373    0.9275    0.6944    196527
weighted avg     0.9773    0.9296    0.9473    196527

Confusion Matrix (Val):
[[177540  13429]
 [   415   5143]]
Validation ROC AUC : 0.9703
```

Fig. 4. SVM performance data validation.

```
[INFO] Using threshold = 0.3793

=== Validation Report ===
              precision    recall  f1-score   support

           0     0.9982    0.9453    0.9711    190969
           1     0.3339    0.9417    0.4930      5558

    accuracy                         0.9452    196527
   macro avg     0.6661    0.9435    0.7320    196527
weighted avg     0.9794    0.9452    0.9575    196527

Confusion Matrix (Val):
[[180529  10440]
 [   324   5234]]
Validation ROC AUC : 0.9847
```

Fig. 5. CNN performance data validation.

C. Testing Results

The testing performance of the models is presented in Fig. 6–Fig. 8 and summarized in Table II.

TABLE II. TESTING PERFORMANCE COMPARISON

Model	Accuracy	Precision	Recall	F1 Score
Random Forest	97,39	98,60	97,39	97,78
SVM	92,96	97,73	92,96	94,73
CNN	94,52	97,94	94,52	95,75

```
=== Test Report ===
              precision    recall  f1-score   support

           0     0.9995    0.9735    0.9864    190970
           1     0.5197    0.9838    0.6801      5557

    accuracy                         0.9738    196527
   macro avg     0.7596    0.9787    0.8333    196527
weighted avg     0.9859    0.9738    0.9777    196527

Confusion Matrix (Test):
[[185918   5052]
 [    90   5467]]
Test ROC AUC : 0.9982
```

Fig. 6. Random forest performance data testing.

```
=== Test Report ===
              precision    recall  f1-score   support

           0     0.9976    0.9279    0.9615    190970
           1     0.2713    0.9221    0.4193      5557

    accuracy                         0.9278    196527
   macro avg     0.6344    0.9250    0.6904    196527
weighted avg     0.9770    0.9278    0.9462    196527

Confusion Matrix (Test):
[[177208  13762]
 [   433   5124]]
Test ROC AUC : 0.9700
```

Fig. 7. SVM performance data testing.

```
=== Test Report ===
              precision    recall  f1-score   support

           0     0.9995    0.9735    0.9864    190970
           1     0.5197    0.9838    0.6801      5557

    accuracy                         0.9738    196527
   macro avg     0.7596    0.9787    0.8333    196527
weighted avg     0.9859    0.9738    0.9777    196527

Confusion Matrix (Test):
[[185918   5052]
 [    90   5467]]
Test ROC AUC : 0.9982
```

Fig. 8. CNN performance data testing.

D. Discussion

Random Forest consistently outperformed SVM and CNN with the highest accuracy and ROC-AUC, demonstrating excellent capability in separating normal and anomalous behaviours. Its interpretability through feature importance also makes it suitable for enterprise operational environments. SVM showed high anomaly recall but at the cost of higher false positives—a trade-off acceptable in early-stage threat detection where missing anomalies is far riskier.

CNN performed strongly in terms of ROC-AUC (> 98%), suggesting that deep models can effectively capture local nonlinear patterns across features. Threshold optimization using Youden's J proved crucial, as it significantly increased anomaly sensitivity across all models while maintaining acceptable accuracy. Despite the strong performance results, the possibility of overfitting should still be considered, particularly because the models were trained on resampled data and evaluated on a single enterprise dataset. However, the use of separate validation and test sets, together with threshold selection on the validation set only, was intended to reduce this risk and improve the credibility of the reported performance. Overall, the findings show that a machine-learning-based UBA system can be reliably deployed using real operational logs from Indonesian enterprise environments.

The comparison in this study focused on three representative models, namely Random Forest, SVM, and 1D CNN, to balance methodological diversity and implementation practicality. Although additional baselines and more recent deep learning architectures could provide a broader benchmark, the selected models were considered sufficient for establishing an initial comparative evaluation on real enterprise web-application logs [29].

V. CONCLUSION AND FUTURE WORK

This research presents a practical and high-performing insider-threat detection approach built using real user activity logs from a major ICT distribution company in Indonesia. Using Random Forest, SVM, and CNN, paired with an effective imbalance-handling strategy (undersampling and SMOTE), the system achieved excellent performance. The experimental evaluation demonstrated that the Random Forest model achieved the highest detection performance with an accuracy of 97.38% and an ROC-AUC of 99.82%, indicating strong capability in distinguishing anomalous user activities.

Conclusions:

- Real enterprise user activity logs can be effectively used for insider-threat detection.
- Random Forest achieved the strongest and most stable results (ROC-AUC 99.82%).
- Thresholding via Youden's J significantly improved anomaly detection sensitivity.
- The proposed approach is suitable for deployment in Indonesian enterprise operational environments.

Future work recommendations:

- Explore temporal models (LSTM, Transformer) for sequence-based behaviour analysis.
- Integrate the system with SIEM platforms for operational security monitoring.

ACKNOWLEDGMENT

The author would like to express sincere gratitude to Bina Nusantara University for the academic guidance, institutional support, and resources provided throughout the completion of this study. The author also gratefully acknowledges PT Synnex Metrodata Indonesia, where the author is professionally employed, for providing a supportive working environment, practical insights, and valuable exposure that contributed meaningfully to the development and completion of this research. The combination of academic supervision and professional experience played an important role in shaping and strengthening this work.

AUTHORS' CONTRIBUTIONS

This research was conducted collaboratively by the authors. Yosep was responsible for data collection, data preprocessing, model implementation, experimental evaluation, and manuscript preparation. Aditya Kurniawan provided academic supervision, methodological guidance, research validation, and critical review of the manuscript to ensure clarity, scientific rigor, and alignment with research objectives.

DATA AVAILABILITY

The dataset used in this study consists of user activity logs collected from an enterprise web application environment within PT Synnex Metrodata Indonesia. The dataset contains records related to user interactions within internal enterprise systems, including activity timestamps, user identifiers, accessed modules, executed actions, session information, and system response data. Due to confidentiality agreements and the presence of sensitive organizational and user-related information, the raw dataset cannot be publicly shared. However, the dataset structure used in this research includes representative attributes such as user_identifiers, activity_timestamp, module_accessed, action_type, request_method, endpoint_url, ip_address, device_type, browser_information, response_status, and activity_sequence. These attributes were utilized to construct behavioral features for the machine learning models applied in insider threat detection. Although the complete dataset remains restricted, the data preprocessing procedures, feature engineering steps, and model implementation methodologies are comprehensively described in this study to support research transparency and reproducibility.

REFERENCES

[1] Cybersecurity and Infrastructure Security Agency (CISA), "Defining insider threats," 2025. [Online]. Available: https://www.cisa.gov/topics/physical-security/insider-threat-mitigation/defining-insider-threats

[2] A. Aryal and A. Shrestha, "User behavior analytics for insider threat detection: A machine learning approach," in *Proc. 10th IOE Graduate Conf.*, Tribhuvan University, pp. 303–310, 2023.

[3] B. Bin Sarhan and N. Altwaijry, "Insider threat detection using machine learning approach," *Applied Sciences*, vol. 13, no. 1, p. 259, 2022.

[4] P. S. Prasad, S. K. Nayak, and M. V. Krishna, "Enhanced insider threat detection through machine learning approach with imbalanced data resolution," *Journal of Theoretical and Applied Information Technology*, vol. 102, no. 3, pp. 914–925, 2024.

[5] X. Ye, F. Luo, H. Cui, J. Wang, X. Xiong, W. Zhang, and W. Zhao, "Research on insider threat detection based on personalized federated learning and behavior log analysis," *Scientific Reports*, vol. 15, no. 1, p. 19214, 2025.

[6] F. R. Alzaabi and A. Mehmood, "A review of recent advances, challenges, and opportunities in malicious insider threat detection using machine learning methods," *IEEE Access*, vol. 12, pp. 30907–30927, 2024.

[7] C. C. Aladi, "Web application security: A pragmatic exposé," *Digital Threats: Research and Practice*, vol. 5, no. 2, 2024.

[8] SaaS Alerts, "The role of user behavior analysis in cybersecurity," 2023. [Online]. Available: https://saasalerts.com/the-role-of-user-behavior-analysis-in-cybersecurity/

[9] A. Ali, M. Husain, and P. Hans, "Real-time detection of insider threats using behavioral analytics and deep evidential clustering," *arXiv preprint*, arXiv:2505.15383, 2025.

[10] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection for insider threat detection using machine learning approaches," *ACM Computing Surveys*, vol. 54, no. 6, pp. 1–36, 2021.

[11] OWASP, "OWASP top 10 web application security risks," 2021. [Online]. Available: https://owasp.org/Top10/

[12] I. J. Vourganas and A. L. Michala, "Applications of machine learning in cybersecurity: A review," *Journal of Cybersecurity and Privacy*, vol. 4, no. 4, pp. 972–992, 2024.

[13] X. Wang, T. Zhang, and M. Qi, "A comprehensive review on machine learning techniques for insider threat detection," *IEEE Transactions on Cybernetics*, vol. 53, no. 1, pp. 329–345, 2023.

[14] F. M. Prabowo, I. Widya, and H. Susanto, "Performance evaluation of machine learning algorithms for insider threat detection based on system logs," *Information Sciences*, vol. 622, pp. 453–471, 2024.

[15] S. Rasheed, M. A. Khan, A. Alghamdi, and A. Alzahrani, "Insider threat detection using behavioural analysis through machine learning classifiers," *IEEE Access*, vol. 10, pp. 98209–98218, 2022.

[16] S. Kim and I. Lee, "LSTM autoencoder-based insider abnormal behavior detection using de-identified data," in *Proc. 10th Int. Conf. on Information Systems Security and Privacy (ICISSP)*, pp. 609–620, 2024.

[17] OpenText, *Advanced insider threat detection with user behavior analytics*, OpenText, 2025.

[18] W. Eberle, L. Holder, and D. Cook, "Insider threat detection based on behavior analysis techniques: A systematic review," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3087–3103, 2021.

[19] I. Hussain, S. U. Khan, and S. H. Gillani, "Insider threat detection in organizational networks using machine learning approaches: A comprehensive review," *IEEE Access*, vol. 8, pp. 15176–15196, 2020.

[20] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 180, p. 103023, 2022.

[21] T. Al-Shehari, M. Al-Razgan, T. Alfakih, R. A. Alsowail, and S. Pandiaraj, "Insider threat detection model using anomaly-based isolation forest algorithm," *IEEE Access*, vol. 11, pp. 118170–118185, 2023.

[22] D. Ojo, M. Al Mhiqani, H. Al Aqrabi, and T. Al Shehari, "Evaluation of machine learning algorithms and SMOTE for insider threat detection," in *Proc. Int. Symp. on Intelligent Computing Systems*, pp. 303–318, 2025.

[23] M. Singh, B. M. Mehtre, S. Sangeetha, and V. Govindaraju, "User behaviour-based insider threat detection using a hybrid learning approach," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 4, pp. 4573–4593, 2023.

[24] M. Elbasheer, "Enhancing insider threat detection using user-based sequencing and transformer encoders," Ph.D. dissertation, The George Washington University, Washington, DC, USA, 2024.

[25] M. Elbasheer and A. Akinfaderin, "User-based sequential modeling with transformer encoders for insider threat detection," *arXiv preprint*, arXiv:2506.23446, 2025.

[26] S. Kushchi, "The in-depth guide to OWASP's top 10 vulnerabilities and how to prevent them," 2025. [Online]. Available: https://www.jit.io/resources/security-standards/the-in-depth-guide-to-owasps-top-10-vulnerabilities

[27] Open AppSec, "Top 10 insider threat detection software," 2025. [Online]. Available: https://www.openappsec.io/post/top-10-insider-threat-detection-software/

[28] G. Sharma, A. Verma, and V. Kumar, "Deep learning models for insider threat detection: An empirical study using CNN and LSTM," *Computers & Security*, vol. 116, p. 102809, 2024.

[29] J. Yi and Y. Tian, "Insider threat detection model enhancement using hybrid algorithms between unsupervised and supervised learning," *Electronics*, vol. 13, no. 5, p. 973, 2024.