**INSIDER THREAT DETECTION IN AN ORGANISATION INFORMATION SYSTEM**

*Major Project Report*

*submitted in fulfilment of the requirement for the Degree.*

*of*

**Bachelor of Technology**

**(Electronics and Communication Engineering)**

***By***

**Poluparthi Supriya (2004118)**

*Under the Supervision*

*Of*

**Dr. Rajan Agrahari Mr.Rohit Kumar**

Assistant Professor **Mr.Niranjan Kulkarni**

Electronics & Communication Reporting Managers

Department Talent Battle

******

**Department of Electronics and Communication Engineering**

**NATIONAL INSTITUTE OF TECHNOLOGY PATNA**

**Patna, Bihar, India-800005**

**Department of Electronics and Communication Engineering**

**NATIONAL INSTITUTE OF TECHNOLOGY PATNA**

**Patna, Bihar, India-800005**

******

**CERTIFICATE**

This is to certify that **Poluparthi Supriya** (Roll Number- 2004118), a bonafide student at National Institute of Technology Patna, has been doing internship of 6 months from January 2024 at **Talent Battle** (ongoing)till June 2024, in partial satisfaction of the requirements for the Bachelor of Technology in Electronics and Communication Engineering for the 2020–2024 academic year at the National Institute of Technology Patna.The project entitled “**INSIDER THREAT DETECTION IN AN ORGANISATION INFORMATION SYSTEM**” meets the criteria for assessment and the granting of a Bachelor of Technology degree, hence it comes highly recommended for approval. All our best to her in her future endeavors.

……………………….. ………………………….

**Dr.Rajan Agrahari Mr.Rohit Kumar**

Assistant Professor Reporting Manager

Electronics & Communication Talent Battle

Department

**Department of Electronics and Communication Engineering**

**NATIONAL INSTITUTE OF TECHNOLOGY PATNA**

**Patna, Bihar, India-800005**

******

**Declaration**

I hereby declare that the work reported in this report entitled “INSIDER THREAT DETECTION IN AN ORGANISATION INFORMATION SYSTEM” submitted at Department of Electronics and Communication Engineering, National Institute of Technology Patna is a bonafide record of our work carried out under the supervision of Mr.Rohit Kumar. I have not submitted this work elsewhere for any other degree or diploma.

Poluparthi Supriya (Roll No – 2004118)

Department of Electronics and Communication Engineering

National Institute of Technology Patna

Date : 30/04/2024

**Department of Electronics and Communication Engineering**

**NATIONAL INSTITUTE OF TECHNOLOGY PATNA**

**Patna, Bihar, India-800005**

******

**ACKNOWLEDGEMENT**

I'd like to thank everyone who supported me during my internship. **Dr. Rajan Agrahari, Assistant**

**Professor**, Department of Electronics and Communication Engineering, National Institute of Technology

Patna & **Mr. Rohit Kumar, Report Manager, Talent Battle** for their perpetual valuable guidance, and

support throughout the internship period, because of which I was able to learn so much more and this will be

of great assistance to me in the future. Since January 2024, I have experimented with a multitude of new

technology platforms, and the learning curve has been precipitous. I would also like to extend my gratitude to **Dr. P.K. Jain**, **Director of the NIT Patna**, for allowing us to participate in a semester-long internship.

Poluparthi Supriya(Roll no- 2004118)

Major Project

Department of Electronics and Communication Engineering.

NIT Patna

**INTERNSHIP DETAILS**

COMPANY: TALENT BATTLE

ROLE: ML Intern



Talent Battle provides web portfolios and various tech projects to various industries and tech companies.It is a Edu- tech startup.We also organize purely online competitions in the field of education to eliminate the barrier of time, money & travel for students willing to battle with their talent on the national platform. We currently also facilitate an android app which is India's First Multiplayer Trivia Game! With a question bank of 60000+ questions from 100+ categories. Users can practice and also challenge their friends and other talented minds across the globe.

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| S.NO. | TOPIC | Page No. |
| 1. | Abstract | 4 |
| 2. | Introduction | 5 |
| 3. | Literature Review | 8 |
| 4. | Ideation of Suitable Model | 10 |
| 5. | Theory | 12 |
| 6. | Methodology | 15 |
| 7. | Implementation | 16 |
| 8. | Result Analysis and Explanation | 17 |
| 9. | Conclusion | 20 |
| 10 | Future Work | 21 |
| 11. | References | 22 |
| 12. | Learning From Internship | 23 |

1. **ABSTRACT**

In the current intranet environment, information is becoming more readily accessed and replicated across a wide range of interconnected systems. Anyone using the intranet computer may access content that he does not have permission to access. For an insider attacker, it is relatively easy to steal a colleague’s password or use an unattended computer to launch an attack.

In this paper, we attempt to discover insider threat by identifying abnormal behavior in enterprise social and online activity data of employees. To this end, we process and extract relevant features that are possibly indicative of insider threat behavior. This includes features extracted from social data including email communication patterns and content, and online activity data such as web browsing patterns, email frequency, and file and machine access patterns.

We start by gathering and organizing data about how people in an organization usually behave. We then mark this data to show what's normal and what might be suspicious. With the help of Keras, we design a computer program that learns from this data.

This program is like a detective. It learns what normal behavior looks like and can tell when something unusual happens.We fine-tune the program to work its best by adjusting settings that make it smarter over time.

To know if our program works well, we give it some new data it hasn't seen before and see how accurate it is at spotting unusual behavior. We use measurements like accuracy and other numbers to decide if our program is doing a good job.

The best part is that once we've trained our program, we can use it to watch people's actions in real time. It's like having a security guard that never gets tired and can quickly point out if something doesn't seem right.

The research focuses on designing, implementing, and evaluating a robust insider threat detection model using Keras. The approach encompasses data preprocessing, model architecture design, training, and evaluation stages, all within the context of the Keras ecosystem.

In summary, this research is about making a program that can help organizations detect insider threats using Keras, a special tool that understands patterns. By using this program, organizations can keep their information safer from potential risks caused by people working within the organization.

*Keywords* :*Enterprise social data, Insider threat, cybersecurity, Keras, behavior analysis, model deployment*

## 2.INTRODUCTION

Insider threats have always been one of the most severe challenges for intranets with security requirements, because they can cause system destruction, information exfiltration, etc. In recent years, with the frequent occurrence of insider threat events, intranet security has aroused increasing attention.

Internal personnel have access to use internal proprietary systems, and they know internal security policies and protection techniques and review regulations from safety facilities, e.g., firewalls and IDS. Hence, internal personnel can bypass existing security facilities. Even worse, a malicious insider may also be the one who configures security measures. Moreover, a cyberattack from insiders is prevalent within the organization; according to the survey, 27% of all cybercrime incidents were suspected to be committed by insiders. There are two reasons for these insider threats; on the one hand, employees with malicious intentions modify or steal organization’s confidential information, trade secrets, or customer data for personal interests. For example, insiders make use of sensitive information to obtain commercial interests or sell them to foreign organizations. On the other hand, internal employees with inadvertent behavior expose the organization’s key assets and sensitive information to external opponents. Furthermore, in some confidential organizations, the insider threat attacks may even be spy activities at the national level. Thus, the effective detection methods are worth studying.

This report investigates the application of the 1D Convolutional Neural Network (1D- CNN) architecture to the problem of insider threat detection using the "sessionr4.2\_binary" dataset. The objective of this study is to design, implement, and evaluate a machine learning model that effectively identifies patterns indicative of insider threats within user activities.

### Problem Statement

The primary focus of this project is to address the problem of insider threat detection. Insider threats involve authorized users misusing their privileges for malicious purposes, making their identification challenging. Machine learning, particularly deep learning, provides an avenue to model complex relationships within user behavior data and identify suspicious activities.

### Motivation

The motivation behind this project is the increasing need for advanced techniques to detect insider threats in organizations' digital ecosystems. Traditional rule-based systems struggle to capture nuanced behaviors, making machine learning an appealing alternative. The 1D-CNN architecture, originally designed for time series analysis, holds promise in identifying patterns within sequences of user activities.

### Dataset: "sessionr4.2\_binary"

The "sessionr4.2\_binary" dataset is a benchmark dataset commonly used in insider threat detection research. It contains a diverse set of features related to user logon sessions, network interactions, file activities, and more. Each instance is labeled as benign or representing an insider threat. The dataset's complexity and real-world relevance make it suitable for evaluating the effectiveness of the 1D-CNN model.

We have utilized the insider threat dataset published by CERT(Computer Emergency Response Team) Carnegie Mellon University for this research. The CERT dataset is not not real-world enterprise data, but it is an artificially generated dataset created for the purpose of validating insider-threat detection frameworks. The dataset “sessionr4.2\_binary” has been used for this analysis.

The CERT dataset is not real-world enterprise data, but it is an artificially generated dataset created for the purpose of validating insider-threat detection frameworks. This dataset consists of eight broad types of data records(session details, information about working hours, employee details, logon/logoff details, file, device, email and HTTP)

1. **Session details :** Session can be defined as a period between a single logon and logoff. Each session is given a session ID and the actions being performed in that particular session are recorded.
2. **Working hours:** To identify the possible anamolies, details such as the day of the session(whether week day/not, working hours/not, before/after working hours are all recorded, duration of work) are all recorded.
3. **Employee details:** Each employee may have different levels of access for the office resources. So keeping track of who is accessing what kind of files is essential in identifying a malicious user. So the employee details like their role, department in which they are working, team, whether/not he/she is an IT admin, etc. are recorded.
4. **Logon/logoff details:** consists of user logon/logoff activities with the corresponding PC with timestamps. ”Logon” and “Logoff” are two kinds of activities can be found in data records. “Logon” activity corresponds to either a user login event or a screen unlock event, while the “Logoff” event corresponds to user logoff event.
5. **Files:** Details of file copies are stored with date, user, PC, filename and content. Accessed files, file transfers are recorded. Has information about file length, file depth, nwords etc.
6. **Device:** This has data records of removable media usage. It indicates insert/remove actions with relevant user,PC and timestamps. Has information about USB connections like the duration.
7. **Email:** A user’s daily e-mail usage logs (number of sent and received e-mails) are stored. It has the details like the sender, receiver, bccs, attachments,mail size. Not just the number of mails, it is sometimes more important to analyze the content of each e- mail than to rely on simple statistics. Individual e-mail-level content analysis is done using topic modeling, which transforms a sequence of words (e-mail body) to a fixed size of numerical vectors to be used for training the insider-threat detection models.
8. **HTTP:** HTTP records contain user, PC, URL and web page content with time stamps.

CSV files facilitate preprocessing, feature extraction, and model training, making them versatile tools in the insider threat detection pipeline. Machine learning and deep learning libraries readily support CSV file import, enabling seamless experimentation and evaluation of the proposed approach.

## 3.LITERATURE REVIEW

### Existing Research work

The detection of insider threats is a critical concern for organizations aiming to safeguard their digital assets. Existing literature emphasizes the importance of identifying individuals who exploit their authorized access for malicious purposes. Traditional approaches often rely on rule-based systems and anomaly detection, but these methods struggle to handle the complexities of nuanced behaviors.

In their work, proposed a comprehensive framework that combines behavioral analysis, machine learning, and graph-based techniques to detect insider threats. They leverage user activity logs and network interactions to build profiles and detect anomalies in user behaviors.

### Deep Learning for Cybersecurity

Deep learning techniques have shown remarkable success in various domains, including natural language processing, computer vision, and speech recognition. In cybersecurity, deep learning models have gained traction for their ability to extract intricate patterns from large datasets.

Several studies have explored the application of deep learning in cybersecurity tasks. They introduce a deep autoencoder-based approach for intrusion detection in network traffic. By learning compact representations of network activities, their model detects anomalies indicative of potential attacks.

### Existing published works

1. presented a statistical approach to detect abnormal behaviors in enterprise's social and online activities. They used different social features such as emails, files, browsing patterns and machine access patterns. They extracted several statistical features to model the abnormality. They used the Vegas dataset with some artificially inserted data from CERT division. They used the iForest method to build their model. Their evaluation ROC score is 0.77.
2. proposed an insider detection model that learned the users' normal behavior and detected the abnormality based on that. They used a Hidden Markov Model on events' logs of normal sequences, for the first five weeks, and used the learned model to detect the possible insider behavior for the rest of the time. They proposed an inertia concept. They used it as an interpolate factor between their new model and the traditional model.
3. proposed a graph based framework for isolation of malicious users based on graphical and anomaly detection techniques. Data from multi dimensional sources of an enterprise network is formatted and fed into the GPU(Graph Processing Unit),

which generates a graph which represents interrelationships between informational assets of the network.

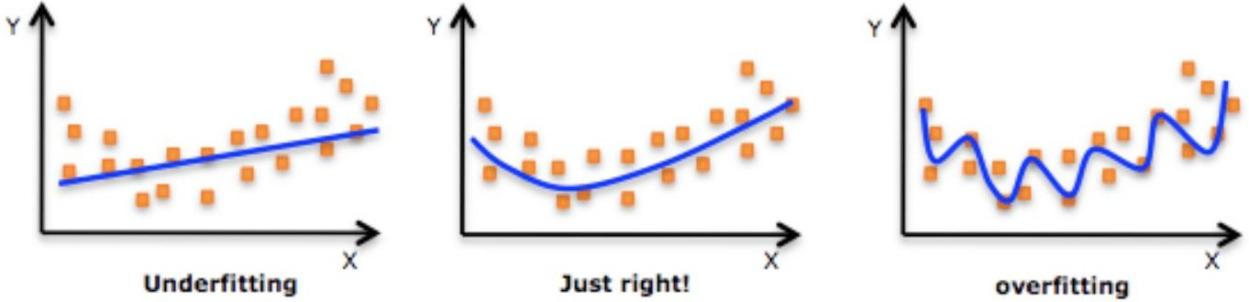
1. research focused on user behavior based insider detection. This paper proposed a DL based approach to detect insiders with higher accuracy and low false positive rate. “LSTM-Auto encoder” has been built on a multi-variate time series data for insider attack detection. They used the version 4.2 dataset for modelling. It is a ‘dense needle’ dataset and it consiseted of total of 32,770,220 rows.The model achieves a remarkable accuracy = 90.60%, precision = 97%, F1Score = 94% and FPR = 9%.
2. proposed a hybrid model combining the potential of Generative Adversarial Networks(GANs) and Bayesian Neural Networks(BNNs) for improved anomaly detection. They proposed a modified Auxiliary Classifier Generative Adversarial Network (ACGAN), referred to as SPCAGAN.
3. proposed an unsupervised deep learning system to filter system log data for analyst. They used Deep Neural Networks (DNNs) and Recurrent Neural Networks (RNNs) to train and recognize activity that is characteristic of each user and judge if it is normal or anamolous. They used CERT dataset version 6.2 and it achieved a recall of 93.5%.

## 4.IDEATION OF SUITABLE MODEL

Depending on the needs and the target to be achieved, different data modelling techniques can be picked.

### Requirements for a good data model

* + - **Accuracy scores:** The chosen model have to be able to give the target results with maximum possible and reliable accuracy scores (An accuracy score of around 87-97% can be considered as a good accuracy score)
    - **Overfitting :** The model should be able to work well not only with the training data, but with the new test data as well.
    - **Underfitting:** Underfitting is a scenario where a data model is unable to capture the relationship between the input and output variables accurately, generating a high error rate on both the training set and unseen data. Underfitting has to be avoided.



* + - **Number of features:** Certain models impose a restriction on the number of features in the data set. So care has to be taken while choosing the model and in case the number of features are more, either feature selection techniques have to be applied or another relevent model has to be chosen.

### Some possible data model options to choose for detecting insider:

1. **Isolation Forest:** Useful for identifying anomalies within data. It doesn't assume any specific distribution of data and can handle high-dimensional data. But, Isolation Forest can work well for simpler cases of anomaly detection only.
2. **Long Short Term Memory Networks (LSTMs):** LSTMs retain information over time. They are useful in time-series prediction because they remember previous inputs.

If the features you have extracted from the data don't provide substantial sequential information, using LSTM might not yield significant results.

1. **Hidden Markov Model :** is a statistical model that is used to describe the probabilistic relationship between a sequence of observations and a sequence of hidden states. It is used to predict future observations or classify sequences, based on the underlying hidden process that generates the data.

HMMs can face challenges when dealing with long sequences or sequences with varying lengths. Insider threat detection data might involve varying

session lengths, which could impact the model's performance.

1. **Convolutional Neural Networks(CNNs) :** 1D-CNNs can capture temporal dependencies in sequential data, making them suitable for analyzing user behavior patterns over time.
2. **Autoencoders:** These unsupervised models can learn to represent normal patterns in the data and flag anomalies by identifying deviations from these patterns.

Autoencoders are often used for anomaly detection where deviations from learned patterns are flagged. If your project's main objective was classification (identifying insider vs. non-insider behavior), classification-focused models like 1D-CNNs might be more suitable.

### 1D-CNN in Insider Threat Detection

The utilization of 1D-CNN architectures for insider threat detection is a relatively nascent area of research. However, the ability of 1D-CNNs to analyze sequential patterns makes them promising candidates for detecting malicious behaviors within user activities.

## 5.THEORY

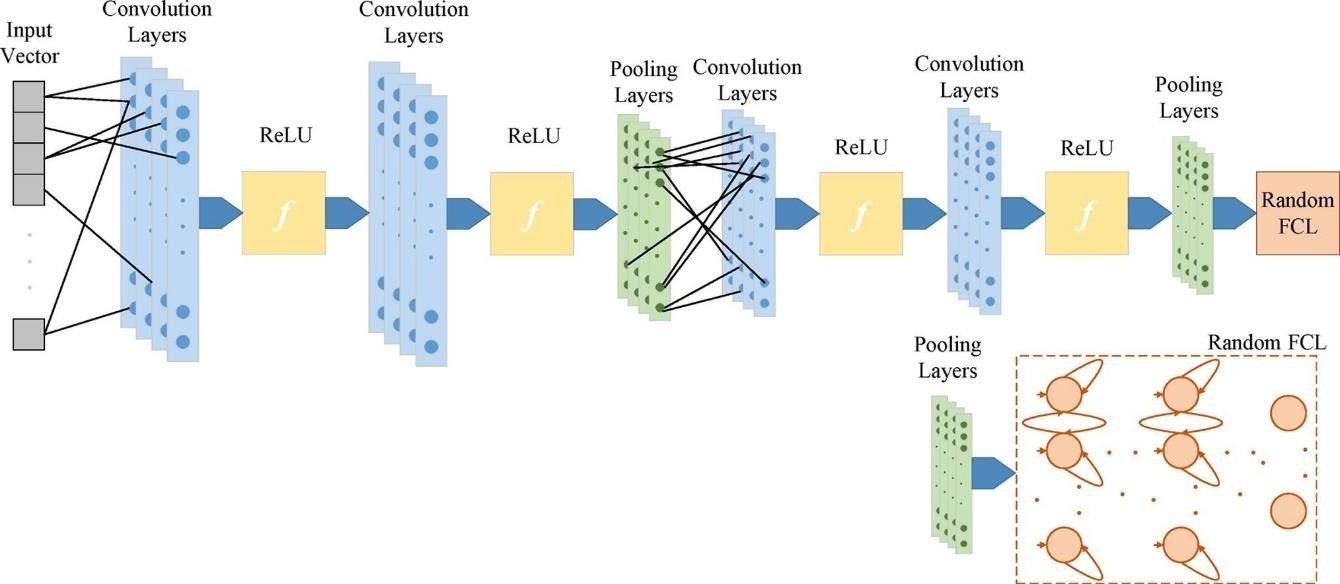
### CNN:

A Convolutional Neural Network is a type of deep learning model inspired by the visual processing mechanism of the human brain. It consists of layers that automatically learn hierarchical features from input data, making it well-suited for tasks involving spatial data like images. By automating the process of feature extraction, CNNs enhance organizations' capacity to identify potential risks originating from within their ranks.

In the context of Insider Threat detection, CNNs play a pivotal role in leveraging their ability to uncover complex patterns within visual data, enabling organizations to identify potential risks originating from within. Insider threats involve identifying individuals within an organization who misuse their privileges to compromise security or engage in malicious activities.

### 1D Convolutional Neural Network (1D-CNN) Architecture

The 1D-CNN architecture was chosen for its suitability in handling sequential data, making it an ideal candidate for insider threat detection based on session activities. The architecture comprises the following components:



**1D Convolutional Layers:** The initial layers use 1D convolutions to capture local patterns and features within sequences of activities. Filters of varying sizes are used to extract different levels of information.

**Max Pooling Layers:** Max pooling layers are applied to downsample the extracted features, reducing the model's sensitivity to minor variations in the sequence.

**Flatten Layer:** The features extracted by the convolutional and pooling layers are flattened into a vector to be passed on to the fully connected layers.

**Fully Connected Layers:** These layers enable the model to learn global patterns and relationships within the data. They culminate in the output layer that predicts the likelihood of an activity being indicative of an insider threat.

### Keras:

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. Keras provides a more straightforward and easier way to define and manipulate neural networks, making it an excellent choice for beginners and for projects that require fast prototyping.

It's often used as a front-end framework for building and experimenting with various machine learning models. Keras allows you to define models using a user-friendly API, making it easier to create complex neural networks without delving into the low-level details. It can run on top of various backend engines such as TensorFlow, Theano, and Microsoft Cognitive Toolkit.

* 1. **Pandas:**

Pandas' contribution to Deep Learning within Insider Threat is so huge. From the initial data loading and cleaning stages to the final model preparation, Pandas simplifies data manipulation, analysis, and transformation.

Its functionalities support efficient preprocessing, insightful exploratory analysis, and seamless data preparation for deep learning models. As organizations continue to combat insider threats, Pandas remains an essential tool for enhancing the effectiveness and accuracy of these projects, ultimately safeguarding valuable resources and sensitive information.

### Tensorflow:

TensorFlow's contributions to Insider Threat projects are substantial, encompassing model architect ture, data preprocessing, training, customization, deployment, and scalability. Its versatile features and libraries empower organizations to build and deploy deep learning models that effectively identify insider threats. As organizations continue to focus on securing their sensitive information and assets, TensorFlow remains an indispensable tool in the fight against insider threats, ensuring data protection and organizational security.

### SMOTE:

Insider threats pose a significant challenge for organizations seeking to protect their sensitive data and resources. Detecting these threats is a complex task, especially when

dealing with imbalanced datasets where instances of malicious behavior are scarce.

The Synthetic Minority Oversampling Technique (SMOTE) has emerged as a powerful tool in addressing this issue. However, detecting these threats is complicated by the fact that genuine behavior vastly outweighs malicious behavior, resulting in imbalanced datasets where positive (threat) instances are rare. The Synthetic Minority Oversampling Technique (SMOTE) is a resampling technique designed to balance class distribution in imbalanced datasets. It achieves this by generating synthetic samples for the minority class (insider threats) based on existing instances. SMOTE offers several critical advantages when applied to insider threat detection.

The Synthetic Minority Oversampling Technique (SMOTE) has become a game- changing approach in addressing the challenges of imbalanced data in insider threat detection. By generating synthetic instances of the minority class, SMOTE helps in creating balanced datasets, improving the accuracy and effectiveness of models in identifying potential threats. As organizations continue to prioritize the security of their resources and data, SMOTE's integration into insider threat detection workflows represents a significant step toward more robust and reliable threat detection systems

## 6.METHODOLOGY

### Data Preprocessing

The "sessionr4.2\_binary" dataset provided by CERT Carnegie Mellon University is the foundation of this research. As the dataset is artificially generated for insider- threat detection validation, it contains a rich variety of features across different categories. To prepare the dataset for model training and evaluation, we performed the following preprocessing steps:

**Data Cleaning:** The dataset was inspected for missing values, and any records with incomplete information were either imputed or removed, depending on the significance of the missing data.

**Feature Selection:** Given the extensive feature set, we carefully selected relevant features for the insider threat detection task. Features from session details, working hours, employee details, logon/logoff details, files, devices, emails, and HTTP interactions were chosen based on their potential to contribute to threat detection.

**Feature Engineering:** For certain features like logon and logoff timestamps, we calculated session durations and working hours to capture temporal patterns.

Additionally, we transformed categorical variables into numerical representations using techniques like one-hot encoding.

### Model Implementation

The 1D-CNN architecture was implemented using the Keras library, which provides a high-level interface for building and training deep learning models. We defined the architecture using Keras' sequential model API, specifying the layers, activation functions, and optimizer.

### Evaluation Metrics

To evaluate the performance of the 1D-CNN model, we employed a set of standard metrics including accuracy, precision, recall, F1-score, and area under the Receiver Operating Characteristic curve (ROC-AUC). These metrics provide a comprehensive view of the model's ability to detect both benign and malicious activities.

## 7.IMPLEMENTATION

**Importing Libraries:** Necessary libraries like TensorFlow, Keras, NumPy, Pandas, Scikit-learn, and imbalanced-learn (imblearn) are imported. These libraries are essential for building and training machine learning models, handling data, and evaluating performance.

**Loading and Understanding Data:** The dataset is loaded using Pandas, and you check the distribution of the target class using the value\_counts() function. The shape of the dataset is also printed.

**Data Splitting:** The dataset is split into training, testing, and validation sets using the train\_test\_split() function. This ensures that the model is trained on one subset of data and evaluated on independent subsets.

**Data Balancing with SMOTE:** The Synthetic Minority Over-sampling Technique (SMOTE) is applied to balance the imbalanced dataset. This is done using the fit\_resample() function from imblearn's SMOTE module. The counts of the target class before and after applying SMOTE are printed.

**Creating and Compiling the Model:** A simple 1D Convolutional Neural Network (CNN) model is defined using Keras. It consists of Conv1D layers followed by a Flatten layer, a Dropout layer, and an output Dense layer. The model is compiled with the Adam optimizer and binary cross-entropy loss.

**Model Training:** The model is trained using the fit() function. Training is performed using the training data and validated using the validation data. The training history is stored in the history variable.

**Model Prediction and Evaluation:** The model's predictions are obtained using predict() and then thresholded to get binary predictions. A confusion matrix is computed using Scikit-learn's confusion\_matrix() function, and a heatmap of the confusion matrix is plotted using Seaborn.

## 8.RESULT ANALYSIS AND EXPLANATION

### Confusion Matrix Heatmap Analysis

1. After Data Collection and preprocessing,

In 1D CNN model, we use im-balancing data as it is to know the use of balancing techniques.

1. After Data Collection and preprocessing, In 1D CNN model,

There are :

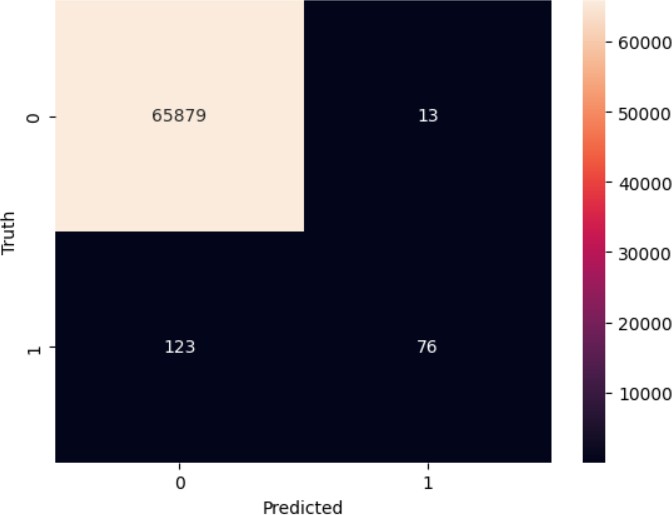
* 1. Input Layer
  2. Hidden Layers
  3. Output Layer

We convert 2D dimension to 3D dimension by reshape method and the Hidden Layers we use are 32 filters with kernel size=5, 64 filters with kernel size=3, 64 filters with kernel size=5 and 128 filters with kernel size=3 and done the max Polling. After that we flatten it and done a Dropout of 0.05.



### Results:

**Confusion matrix:**



True Positives : 65879 False Positives: 13

True Negatives: 76 False Negatives:123

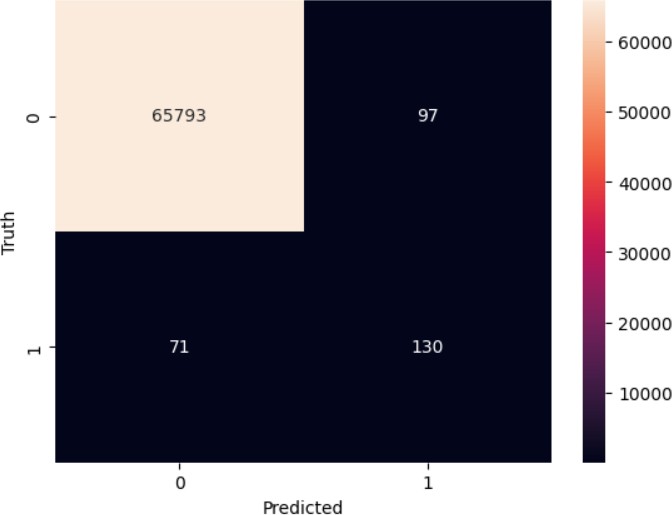
Accuracy : 99.79422311661195

precision : 85.39325842696628

recall : 38.19095477386934

f1.score : 52.77777777777778

2. `+Balanced\_accuracy : 69.08561275996934



True Positives : 65793 False Positives: 97

True Negatives: 130 False Negatives:71

Accuracy : 99.74580502640299

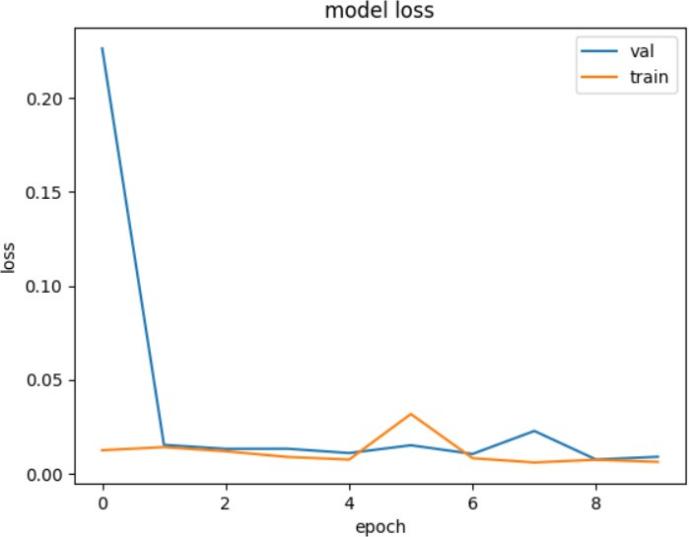
precision : 57.268722466960355

recall : 64.6766169154229

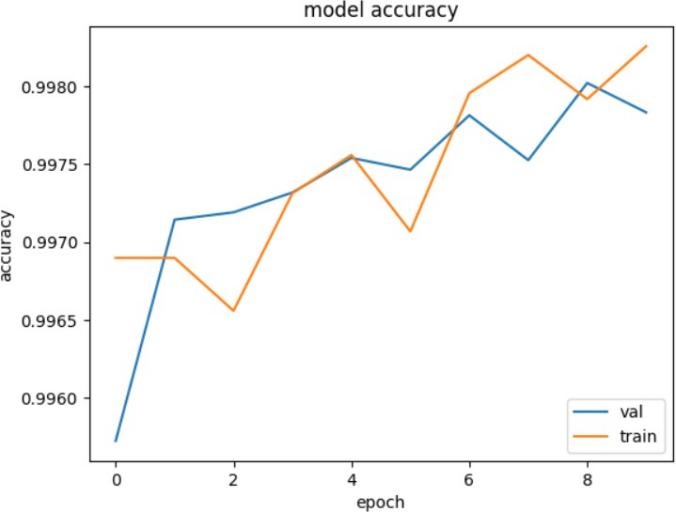
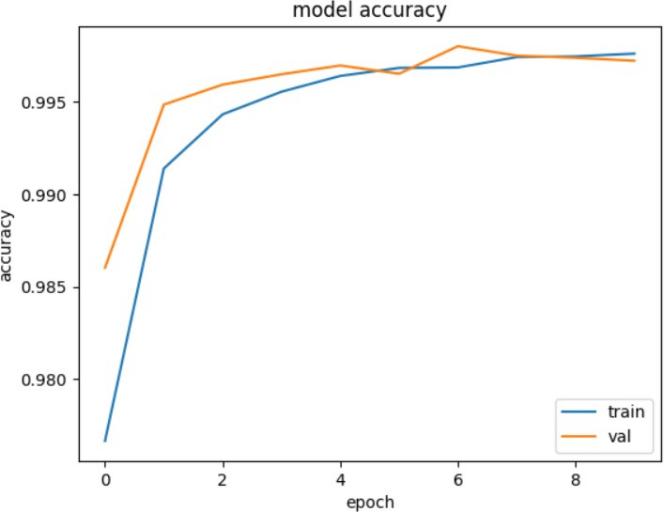
f1.score : 60.74766355140188

Balanced\_accuracy : 82.26470093001377

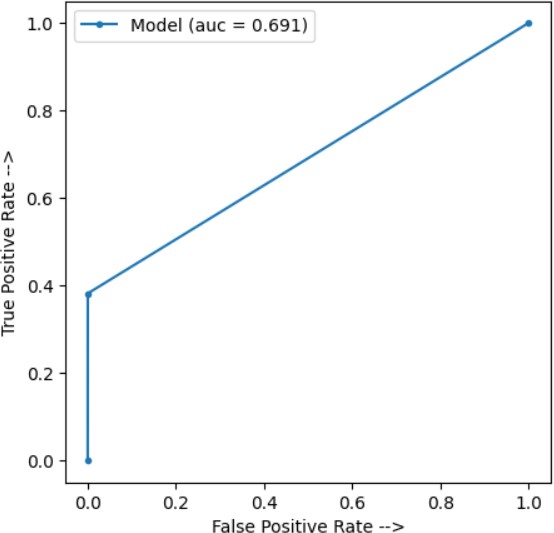
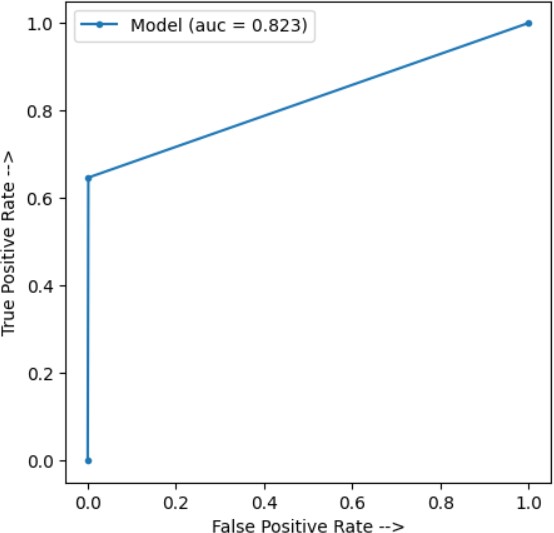
### A graph with blue and orange lines Description automatically generatedLoss vs. epochs:



**Accuracy vs. epochs** :

### ROC curve:



**8.2.precision, recall and f1-score insights**

**Precision:** Evaluate the precision of the model, which is the ratio of true positives to the total predicted positives (TP / (TP + FP)). Higher precision means fewer false positives, indicating that when the model predicts a threat, it's likely to be accurate.

**Recall (Sensitivity):** Assess the recall, which is the ratio of true positives to the total actual positives (TP / (TP + FN)). Higher recall indicates that the model is effectively capturing a significant portion of actual insider threats.

**F1-Score:** The F1-score is the harmonic mean of precision and recall, providing a balanced measure that considers both false positives and false negatives. A high F1- score suggests that the model is successfully balancing precision and recall.

### 9.CONCLUSION

The initial section of the code introduces a technique aimed at managing class imbalance using Synthetic Minority Over-Sampling Technique (SMOTE). This strategy is particularly valuable for enhancing the model's efficiency when handling datasets characterized by imbalanced class distributions. SMOTE functions by generating synthetic instances within the minority class, thus mitigating the bias often introduced when a dataset has significantly more instances of one class compared to the other. This leads to a more balanced representation and subsequently aids the model in learning from the minority class, which can be vital for accurate insider threat detection.

Moving forward, the subsequent portion of the code illustrates the implementation of an 1D-CNN model tailored for Insider Threat Detection. 1D-CNN is configured using the following hyper-parameters: number of CNN layers, neurons in each layer, size of the filter, and subsampling factor of each layer. The convolution layer is the basic application of the filter to an input. Utilizing the filtering operation repeated times creates a feature map, which indicates the particular attributes related to the data points.

Convolution is a linear operation that involved containing the multiplication with inputs with a set of weights. The synergy between these techniques constitutes a comprehensive approach. By first addressing data imbalance through SMOTE and subsequently employing CNN models, the project endeavors to significantly enhance the identification of insider threats. This amalgamation optimally equips the model to tackle the challenges posed by imbalanced datasets and exploit the inherent temporal patterns in user behavior logs.

This research journey has not only enhanced our understanding of insider threat detection but also highlighted the continuous evolution of cybersecurity techniques. By blending advanced machine learning methodologies with domain expertise, we strive to contribute to the ongoing efforts in ensuring secure digital environments.

As we conclude this study, we acknowledge that insider threat detection remains a dynamic field, where ongoing research and collaboration are pivotal for tackling emerging challenges. We hope that our insights inspire further exploration and innovation in the realm of cybersecurity.

Ultimately, this joint strategy aims to bolster the accuracy and reliability of insider threat detection, thereby fortifying the security posture of the system against potential internal risks.

## 10.FUTURE WORK

**Ensemble Methods:** Investigate the effectiveness of ensemble methods such as bagging, boosting, and stacking to combine predictions from multiple models. Ensemble techniques can enhance overall model robustness and generalization.

**Feature Importance Analysis**: Perform a thorough analysis of feature importance to identify the most influential features in insider threat detection. This analysis can guide feature selection, engineering, and help in focusing on the most relevant aspects.

**Temporal Models:** Explore the use of recurrent neural networks (RNNs) or transformers to capture long-range temporal dependencies within sequential data.

These models could provide a deeper understanding of user behavior patterns over time.

**Domain-Specific Pretraining:** Pretrain the model on domain-specific data to capture nuances specific to insider threat detection. Transfer learning from related tasks like anomaly detection could improve model performance.

**Data Augmentation:** Apply data augmentation techniques to artificially increase the diversity of the dataset. This can mitigate the impact of class imbalance and enhance the model's ability to capture various scenarios.

**Real-Time Monitoring:** Develop a real-time monitoring system that continuously analyzes user activities and triggers alerts for potential insider threats as they occur. This requires efficient model deployment and integration with network logs.

**Behavioral Profiling:** Implement user behavioral profiling to create a baseline of normal user behavior. Deviations from this baseline could trigger alerts for suspicious activities.

**Multimodal Analysis:** Integrate information from multiple data sources such as user logs, device usage, emails, and files to create a comprehensive model that considers different aspects of user behavior.

**Adversarial Attacks:** Investigate the model's vulnerability to adversarial attacks that aim to deceive the model's predictions. Adversarial training techniques and robust models can help mitigate this risk.

**Deployment Considerations:** Address practical deployment challenges, such as model integration with existing security systems, scalability, and maintenance in a real-world environment.

**User Context Modeling:** Incorporate user context information, such as role changes, promotions, or organizational shifts, to enhance the model's ability to adapt to changing user behavior.

**Continuous Improvement:** As new data becomes available, continuously update and retrain the model to adapt to evolving user behavior patterns and emerging threat scenarios.

Delving into these areas for future work can contribute to advancing insider threat detection techniques, ensuring organizational security in an ever-changing digital landscape.

## 11.REFERENCES

[1] <https://dl.acm.org/doi/abs/10.1145/2808783.2808784>

[2] <https://dl.acm.org/doi/abs/10.1145/2995959.2995964>

1. <https://arxiv.org/abs/1809.00141>
2. <https://ieeexplore.ieee.org/abstract/document/9559986/>
3. <https://arxiv.org/abs/2203.02855>
4. <https://arxiv.org/abs/1710.00811>

1. <https://link.springer.com/chapter/10.1007/978-3-319-93698-7_4>

* 1. <https://ieeexplore.ieee.org/abstract/document/9559986/>
  2. <https://ieeexplore.ieee.org/abstract/document/9315932/>
  3. [https://www.sciencedirect.com/science/article/pii/S0167404821](https://www.sciencedirect.com/science/article/pii/S0167404821000456) [000456](https://www.sciencedirect.com/science/article/pii/S0167404821000456)
  4. <https://ieeexplore.ieee.org/abstract/document/8778466/>
  5. <https://ieeexplore.ieee.org/abstract/document/9020760/>
  6. [https://machinelearningmastery.com/cnn-models-for-human-](https://machinelearningmastery.com/cnn-models-for-human-activity-recognition-time-series-classification/) [activity-recognition-time-series-classification/](https://machinelearningmastery.com/cnn-models-for-human-activity-recognition-time-series-classification/)

## 12.LEARNINGS FROM INTERNSHIP

**Convolutional Neural Networks (CNNs):** Obtained a great understanding of how CNNs work, their architecture, and their effectiveness in capturing spatial patterns within sequential data, which is particularly relevant for time-series and log data like in insider threat detection.

**Convolutional Layers:** We learnt about the concept of convolutional layers and how they perform feature extraction by applying convolutional filters to input data. These filters capture relevant patterns and help the model identify important information.

**Activation Functions:** We understood the significance of activation functions like ReLU (Rectified Linear Unit) in introducing non- linearity to the model, enabling it to capture complex relationships between features.

**Model Architecture:** Designed and configured the architecture of the 1D-CNN model. This includes specifying the number of layers, types of layers (convolutional, pooling, dense), and the sequence in which they are connected.

**Overfitting Prevention:** Explored techniques to prevent overfitting, such as dropout layers and early stopping. These methods help the model generalize better to unseen data and avoid memorizing the training dataset.

**Loss Functions:** We worked with loss functions, such as binary cross-entropy, which quantify the difference between predicted values and true labels. The model minimizes the loss during training to learn the optimal parameters.

**Optimizer:** We used optimizers like Adam to adjust the model's weights during training, aiming to minimize the loss function and improve the model's predictive performance.

**Batch Processing:** We learnt how the model processes data in batches during training, which enhances training efficiency and enables models to fit into available memory.

**Model Evaluation:** We assessed the model's performance using metrics like accuracy, precision, recall, F1-score, ROC-AUC, and confusion matrices. We understood the significance of these metrics in differentiating between true positives, true negatives, false positives, and false negatives.