Insider Threat Classification Using Computer Visison and Deep Learning

Muhammad Ali
Computer and Information System Engineering
NED University of Engineering and Technology
Karachi, Pakistan
ali4106511@cloud.neduet.edu.pk

***Abstract*— Cybersecurity attacks can arise from internal and external sources. The attacks are initiated by internal sources and are called insider threats. These attacks can cause a serious concern to organizations because of the significant damage that can be inflicted by malicious insiders. Features from the CMU CERT insider threat dataset are extracted as usage patterns of insiders and represented these features as images. Hence, images are used to represent the resource access patterns of the employees within an organization. After the construction of images, Deep Convolutional Neural Network (DCNNs) are used for anomaly detection, with the aim to identify malicious insiders. Experimental results show that this method is effective and outperforms other methods for solving insider threat classification problems including traditional machine learning models such as KNN, and random forest for the classification of malicious insiders.**

Keywords*— *Cybersecurity, Insider threats, Deep learning, Computer vision, Classification, Machine learning

I. INTRODUCTION

Insider threats are one of the most common cybersecurity threats. Employees in the company have access to all the resources and data and it is easy for them to breach as compared to external ones. There are many reasons why insiders turn themselves into threats like a human error caused by negligence, conflict with other workers and employees or a demand from competitive companies to give the information. The reason can be any but the loss to the company in terms of financial or reputation can be great. It was estimated in 2004 that "insiders were responsible for more than 80% of data breach incidents in the last four years" [1]. There are multiple techniques to solve the problem but still, there is a window to find a more efficient solution to the problem.

A recent survey conducted by the FBI has found monetary value of a prosperous attack by insider is significantly greater than foreign attack, with the former being almost 50 times more expensive. This highlights the importance of implementing effective measures to detect and prevent malicious insider actions, which can cause severe financial damage to organizations. Insiders cannot only harm company's information resources, but also can damage to the company's credibility, resulting in financial losses. Larry Knutsen, a member of the Laconia National Security Consulting Group, points out that "privileged users, who have been given exclusive access to data within a company, are a major concern for organizations."

Various techniques can be used to solve the problem such as data analysis to reduce the risk of breaches. The use of statistical learning, machine learning, artificial intelligence, and natural language processing methods has expanded to address cybersecurity use cases such as identifying malware and intrusions, phishing, denial of service (DoS) attacks, etc. Security analytics is a method of processing data that incorporates ways for data collection, collection, and evaluation for threat detection and security monitoring. Large and varied datasets may be used to apply security analytics solutions employing machine learning, deep learning, and AI frameworks. One of the most widely used deep learning models, Deep Convolutional Neural Networks (DCNNs), are quite effective at processing visual input, such as images and video frames.

In the problem of insider threats detection attackers are already aware of the security measures are put in place in the business and are familiar with how and where sensitive data is stored, the issue of insider activities is that they may only leave a minor trace in the data. This is the reason why some insider occurrences take a while to come to light. The difficulty of developing effective and efficient data analysis systems for insider attacks still exists.

The difficulty of detecting insider threats is because of a variety of insider threat types, the such as risk incurred due to human negligence or a user with malicious intent to harm the organization. It is important to collect and analyze massive volumes of resource access log data at the expense of storage and computational complexity. For discrepancy or user behaviour analysis or insider threat classification the data has to be translated into a structured format or important features need to be extracted. One data representation structure can be images as well. In this paper image-based approach is used for solving the problem of insider threat classification. Images can be used to portray the behaviour of employees in the organization or resource usage patterns afterward various deep learning models can be used to find the patterns in the data. Once the patterns are detected any activity log can be classified and detected.

Started with the raw data extracted from a benchmark dataset extracted various features from the dataset and formed feature vectors (1D array) and grayscale images. The feature vector and grayscale image represent the everyday usage of an employee within the organization. Afterward, machine learning techniques such as KNN, and random forest are deep neural network techniques applied to a feature vector and deep convolutional neural network technique on grayscale images. At last, the result of all the techniques was compared ad it was found that DCNN outperformed other models.

II. LITERATURE REVIEW

A recent study by Ponemon titled "Cost of Insider Threats a global report" [2] highlights the growing issue of insider threats. The study found that:

- Sixty percent of companies experienced more than thirty incidents related to insiders per annum.
- Sixty-two percent of cases were due to carelessness.
- Twenty-three percent of cases were caused by insiders having malicious intent.
- Fourteen percent of cases were caused by stolen user credentials.
- The number of incidents related to insiders increased by 47% over two years.
- Organization spent around $755,760 per year on insider related incidents.

Many solutions have been developed by researchers to detect insider threats. In [3] they have categorized the classification methods in multiple categories according to techniques and characteristics used, including role and anomaly related controls, realization of risk with psychological factors, risk analysis using workflow, strengthening network defense, enhancing access control and process control to deter malicious insiders. User profiling, which is a part of behavior analysis, helps identify unusual user behavior and is widely recognized as a key method for detecting insider threats. Given the scarcity of data from companies, researchers often resort to using their own data collection techniques or synthetic data. One example of this is the dataset created by the Computer Emergency Response Team (CERT) at Carnegie Mellon University (CMU), which simulates an organization and includes log files mimicking the actions of employees. In [4] this synthetic data is referred to as the CMU CERT insider threat dataset. Liu in [5] proposed a method of feature extraction from log data. Deep autoencoders are used for unsupervised learning. They can be used to recognize patterns in data, such as malicious insider actions. They consist of an encoder and a decoder that work together to compress and reconstruct input data. The encoder compresses the input data into a lower-dimensional representation, which is called a bottleneck. The decoder then reconstructs the original input data from the bottleneck. To identify malicious insider actions, a group of deep autoencoders can be trained on a dataset of normal behavior. These autoencoders can then be used to detect anomalies in new data that may indicate malicious actions. Bhodia [6] in implemented image-based malware classification using transfer learning and got better results than other machine learning models. Similar approach of image based is used for insider threats by opens a window for implementing grayscale image-based classification as done by Gayathri in [7].

III. METHODOLOGY

Numerous scenarios make it evident that individuals who carry out insider attacks often display specific personality traits, characteristics, and behaviors that can alert other employees to an impending attack. The CMU CERT dataset [4] on insider attacks is standard for identifying insider threats, utilizing a collection of event logs obtained from a simulated company's computer network. These behaviors and the regular usage patterns of insiders in the company can help identify such culprits. The data set includes a variety of log files containing information such as when a user logs in or out, their browsing history, how files are accessed, use of external devices within the organization, and emails sent or received by the user. Features such as count of logins per day outside office hours, emails sent outside domain of organizations are accessed from logs and converted as images in a grayscale mode. After converting those features into images, deep convolutional neural network is created and images are given to it as an inputs. As a result, good classification performance is observed.

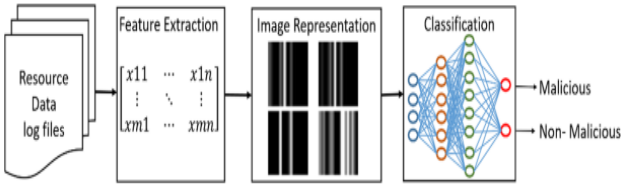

Fig. 1. *Flow diagram of our proposed methodology*

A. Feature Vector Creation

Obtaining data for insider threat detection can be difficult as organizations tend to keep their data private. However, researchers have widely used the CMU CERT dataset as a benchmark for testing their methods. This synthetic dataset simulates various scenarios of malicious activities. The organization's employee resource access patterns are captured and analyzed using log files that include various types of information. These log files include:

- logon.csv: User's logon/logoff details, the system used to login which provides insight into the user's computer access patterns
- files.csv: Information of files accessed by the user with timestamps, also the content of the file from this we can infer the file usage patterns.
- device.csv: Information about the user's external device access on each day, including access during and outside of office hours. This data can be used to identify malicious activity.
- email.csv: It contain details about emails sent to and fro within the domain and outside of domain during and outside of office hours. All the necessary details of emails can be accessed from the file, such as size of email, number of attachments and others.
- http.csv: It contains all the information of sites visited by the user including website contents, the pc used for accessing the sites and timestamps at which the websites were accessed.
- answers.csv: It contains all the malicious logs.

Various features are extracted from these files known as time-based features. After finding the features, data

preprocessing is performed, which includes normalization and random undersampling for handling class imbalance problems. The final feature vector contains all the feature set values on the basis of each user's per day usage. Figure 2 shows the entire process of feature vector construction.

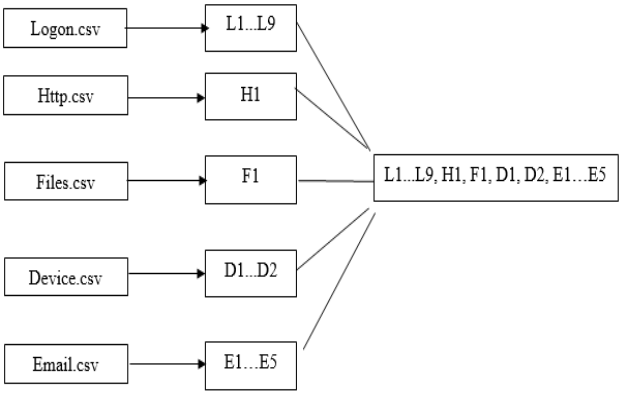

Fig. 2. *Feature Vector creation process*

B. Grayscale image generation

Our problem can be solved by traditional machine learning algorithms like Support Vector Machines, Random Forest, Naïve Bayes and K-Nearest Neighbors. Also, they perform classification and give good results, but to choose the best one, it is essential to try multiple algorithms and compare their performances to select an efficient one for the problem.

DCNNs are a type of neural network that are used for image processing tasks, as they can learn and extract features from images directly. As the dataset does not have any images, so we need to convert tabular data into grayscale images. It is possible to use the same DCNN architecture and techniques to process this data as well. Converting tabular data into grayscale images can help understand the underlying patterns and structures in the data. Here we are not using colored images because using grayscale images as input allows the model to use less computational resources and memory compared to color images, and it is easier to run models on low-powered devices. Images are formed with the approach proposed by Alok in "A methodology to transform a non-image data to an image for convolution neural network architecture" [8]. As we have 17 features, so the minimum possible size of image is 5x5. Figure 3 shows the sample grayscale image.

Fig. 3. *Sample grayscale image*

C. Classification

Classification models are a type of machine learning algorithm applied to forecast the category of an input. These models learn to associate certain inputs with specific classes or labels by analyzing patterns and relationships in a training dataset. Once trained, the model can then be applied to forecast the category of new input. We have 2 classes Malicious and Non-malicious so we will be using binary classifiers. I will use the same dataset for traditional binary classifiers and deep learning-based classifiers.
Two structures of data are used to perform experiments. At first, we will be doing classification with KNN, Random Forest and a simple deep neural network using the feature vector created (1D array) above.

Secondly, Deep Convolutional Neural Network (DCNN) is used for the classification of images. For the categorization of the image data, use a Deep Convolutional Neural Network (DCNN). DCNNs start with raw data as their input and process it from several elementary computational units called neurons to produce representations that are helpful for classification in higher layers. Figure 4 shows the working of DCNNs.

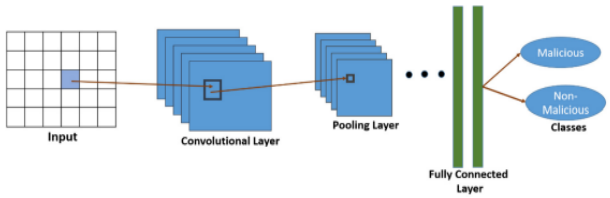

Fig. 4. *Methodology for DCNN*

To determine if a user is an insider or not, the DCNN receives the grayscale image. The prediction is carried out by the network by calculating the probability for each class label. The loss function is used to determine the prediction error. Models of Deep Convolutional Neural Networks are often utilized in image identification and classification. Each input picture is sent through a sequence of convolution layers with filters, pooling, and fully connected layers in order to train and test the images.

IV. Implementation

In this section I will present the details about implementation of different models on the feature vector and image dataset. Firstly, dataset is described, secondly the class imbalance problem and lastly the experimental results. We have used Scikit learn for traditional machine learning models such as KNN and random forest. Keras is used with TensorFlow for the development of deep neural networks.

A. Dataset

Given the scarcity of data from companies, researchers often resort to using their own data collection techniques or synthetic data. One example of this is the dataset created by the Computer Emergency Response Team (CERT) at Carnegie Mellon University (CMU), which simulates an organization and includes log files mimicking the actions of employees. In [4] this synthetic data is referred to as the CMU CERT insider threat dataset.

All the five event files for activities related to login and logoff, http data, email correspondence, file operations, and use of an external storage device. Events from these log files are utilized as input to determine which instances were malicious and which ones were not, which are then used to construct the feature vector and the representative images. Our feature extraction technique separates the numerous characteristics into a feature vector of malicious and non-malicious occurrences. The dataset and multiple files with their own column names and data. Each file has logs related to each user for a specific day. We have 1000 users, and their everyday usage can be extracted from these files. Figure 5 shows the various counts of dataset files and number of instances formed.

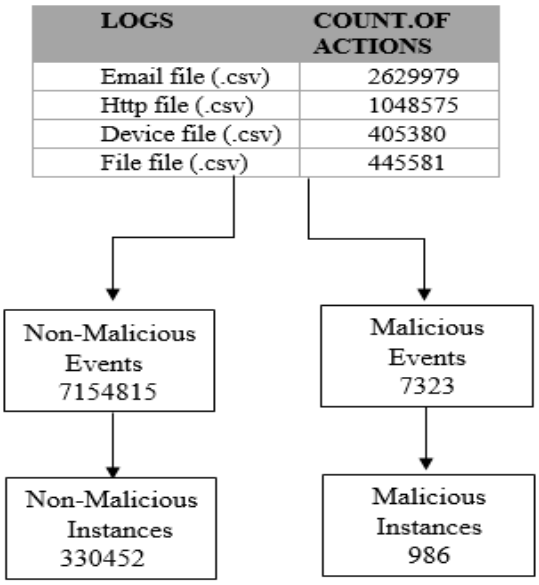

Fig. 5. *Log count in dataset*

Following set of time-based features are extracted from the log files and selected for model training and testing.

TABLE I. LIST OF FEATURES EXTRACTED FROM DATASET

Log File	Features	Description
Logon	L1	Difference between start time of office and first login activity time.
	L2	Difference between office end time and last login activity time.
	L3	Average difference of time between office starts time and number of logins before work hours.
	L4	Average difference of time between office end time and number of logins after work hours.
	L5	Total count of logins
	L6	Count of login after office hours.
	L7	Count of accessed PCs.
	L8	Count of PCs used after work hours.
	L9	Average session duration after work hours.
Email	E1	Number of emails sent on the address outside the organization.
	E3	Count of attachments.
	E4	Average email size.
	E5	Count of receivers.
Device	D1	Count of external device usage
	D2	External devices usage after work hours
File	F1	Count of files with .exe extension downloaded.
Http	H1	Wikileaks.org accesses.

B. Imbalanced data handling

As from the figure it can be seen that the distribution of classes (non-malicious and malicious) is not equal which makes the CMU CERT dataset highly unbalanced. The ratio is 1:340 for malicious and non-malicious instances. This imbalance distribution can negatively impact the model evaluation metrices. In this project, undersampling method to solve the data imbalance problem.

The sampling ratio to reduce the samples from the non-malicious class instances is 5. This results in 4080 non malicious samples and 986 malicious samples. Still, we don't have same number for malicious and non-malicious samples, but this is the optimal ratio to get optimal results.

C. Experimental Results

The CMU CERT v4.2 data are used for the assessment. Images and 1D feature vectors are used to represent the features. To address the issue of the class imbalance, data are sampled randomly. Further train-test split is done with a ratio of 0.3, before applying the feature vectors as an input to the models. Various machine learning models used for evaluations are discussed.

K-nearest neighbours (KNN) algorithm is supervised machine learning algorithm that may be used for both regression and classification, in this case for binary classification of malicious and non-malicious logs. It is used for the classification of 1D arrays and the parameters for KNN that gives the optimal performance are selected such as value of K equal to 5. The results of this classification in terms of classification metrices such as accuracy, precision, recall and f1 score are noted.

TABLE II. CONFUSION MATRIX FOR KNN

Class	Precision	Recall	F1-Score	Accuracy
Non-malicious	0.95	0.96	0.96	0.93
Malicious	0.85	0.81	0.83	

Random Forest binary classifier learns from a collection of labeled instances by training a group of decision trees. The algorithm builds each decision tree throughout this training phase by picking a subset of the data's characteristics at random. Using the patterns that these trees found in the training data, they learned to predict the future. Following results were obtained by giving the feature vectors as an input to the model.

TABLE III. CONFUSION MATRIX FOR RANDOM FOREST

Class	Precision	Recall	F1-Score	Accuracy
Non-malicious	0.96	0.96	0.96	0.93
Malicious	0.83	0.83	0.83	

Deep Neural networks (DNNs) are a type of machine learning technique that mimics the human brain. They are composed of layers of coordinated signals that compute and transfer data. An input layer, one or more hidden layers, and an output layer are commonly found in a deep neural network, a form of neural network with numerous layers. I have taken 17 features so an input layer with having 17 nodes is constructed and then 2 hidden layers are added each time dividing the number of nodes by two and lastly an output layer with one neuron having a sigmoid function is present. The dropout layer is added to avoid overfitting with a rate of 0.5. An output layer with one neuron and activation function of Sigmoid is used.

TABLE IV. CONFUSION MATRIX FOR DNN

Class	Precision	Recall	F1-Score	Accuracy
Non-malicious	0.95	0.96	0.95	0.93
Malicious	0.85	0.79	0.82	

Deep Convolutional Neural Network (DCNN) specifically developed for image and video processing applications; convolutional neural networks (DCNNs) are a subset of deep neural networks. They are trained on labeled image datasets and are made up of several layers of linked neurons that process and send information. The convolutional layer, which typically applies a series of filters to the input image to recognize and extract certain characteristics like edges, textures, and patterns, is a major part of a DCNN. These filters are used to extract information from the image and are learned throughout the training phase. I have used 3 convolutional layers and 2 fully connected layer, with the number of nodes as [32x64x128x64x1]. Sigmoid activation function is used for output layer.

TABLE V. CONFUSION MATRIX FOR DCNN

Class	Precision	Recall	F1-Score	Accuracy
Non-malicious	0.95	0.96	0.95	0.93
Malicious	0.85	0.79	0.82	

In table 6 we have summarized the results of all the models used for classification.

TABLE VI. CONFUSION MATRIX SCORES FOR MALICIOUS CLASS OF ALL APPLIED MODELS

Model	Accuracy	Precision	F1-Score	Recall
KNN	0.93	0.85	0.83	0.81
Random Forest	0.93	0.83	0.83	0.83
DNN	0.93	0.85	0.82	0.79
DCNN	0.93	0.80	0.83	**0.87**

After testing different approaches for solving an insider threat classification, it's important to analyze the results and find an efficient technique for our problem. We will see and analyze the classification report, confusion matrices, and other metrics for comparing models to find an efficient one. By contrasting predicted and actual values, a confusion matrix is a table that is often used to assess how well a classification model performs. A confusion matrix may be used to assess a classifier's performance on an imbalanced dataset when the number of observations in one class differs noticeably from the number of observations made in the other class or classes. When working with unbalanced datasets, it's important to consider the trade-off between precision and recall. A classifier that is optimized for precision will have a low recall and vice versa. Therefore, it's crucial to have a clear understanding of the problem and decide which metric is more important for the specific use case.

In the context of insider threat classification, precision, and recall are two important metrics to evaluate the performance of a classifier. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. A high precision indicates that the classifier has a low rate of false positives, meaning that it rarely flags non-malicious insiders as malicious. Recall, on the other hand, is the ratio of correctly predicted positive observations to the total actual positive observations. A high recall indicates that the classifier has a low rate of false negatives, meaning that it rarely misses actual malicious insiders. In the case of insider threat classification, it's crucial to have a high recall to avoid missing any malicious insider, as the consequences can be severe. For instance, if the log is malicious (Actual Positive) and is predicted as non-malicious (Predicted Negative), then the results will be inappropriate. For this reason, we will select a classifier that has the highest recall score. However, having high precision is also important to avoid falsely accusing non-malicious insiders.
From the above table 6, we can see that all classifiers have the same accuracy, but the recall of DCNN is the highest,

so we will be choosing DCNN as the best classifier for our problem of insider threats classification.

V. DISCUSSION

In short, I started a problem of insider threats and aimed to find an efficient solution, my proposed methodology to apply an image-based problem really outperformed other models and techniques to solve the problem. It ended with 93% accuracy same as others but with a recall of 87% leading other approaches. Some of the questions that arise is why DCNNs outperforms other techniques. So, one reason can be automated feature extraction: DCNNs have the ability to automatically learn and extract features from images, which are more robust. Second reason could be that DCNNs can model non-linear decision boundaries, which are more flexible than linear boundaries and allow them to classify complex images.

Solving every problem using image based approach doesn't always guarantee the best results, it depends on the specific problem and dataset. CNNs can be better than traditional machine learning algorithms for image classification tasks because CNNs are able to learn complex features and patterns from images by using multiple layers of parameters that can be trained. Luckily, we found this approach to be an effective one. Lastly there is still room for improvement such as using Transfer learning and models i.e., VGG16, MobileNet and others.

VI. CONCLUSION

Using traditional classification algorithms on tabular data for the insider threat classification problem is effective, but they have limitations when handling complex and high-dimensional data. These algorithms rely on hand-crafted features and linear decision boundaries, which may not be able to capture the underlying patterns and relationships in the data.

Converting tabular data to grayscale images and then applying DCNNs can be a more efficient approach for insider threat classification. DCNNs are able to automatically learn robust features and model non-linear decision boundaries, which allows them to handle complex and high-dimensional data better. By converting tabular data to grayscale images, the data can be passed through multiple layers of convolutional and pooling layers that extract features from the data, which can then be passed through a fully connected layer to generate the final output. Additionally, DCNNs are able to handle large-scale data, which traditional machine learning classifiers may struggle with. They can be trained on large amounts of data, which results in better generalization and performance, although due to the class imbalance problem, we were not able to deal with millions of logs; otherwise, if the data had been balanced, DCNN would have performed much better.

Started with the CMU CERT v4.2 dataset, extracted features from it, and then various pre-processing operations were performed. The dataset had a significant imbalance between non-malicious and malicious instances, with a ratio of 1:340. To address this imbalance, random undersampling is applied. KNN and DNN models are applied to the data, and results are evaluated. Afterward, to apply DCNN we converted the feature vector into images. The image-based representation showed fairly good precision, recall, and f1-score, even with the undersampling rate of 5. The results found that utilizing an image-based representation can be effective in detecting insider threats.

VII. REFERENCES

[1] H. H. Thompson, J. A. Whittaker, and M. Andrews, "Intrusion detection: Perspectives on the insider threat," Comput. Fraud Secur., vol. 2004, no. 1, pp. 13–15, Jan. 2004, doi: 10.1016/S1361-3723(04)00018-1.

[2] "2022 Ponemon Cost of Insider Threats Global Report|ProofpointUS." https://www.proofpoint.com/us/resources/threat-reports/cost-of-insider-threats (accessed Feb. 13, 2023).

[3] A. Sanzgiri and D. Dasgupta, "Classification of Insider Threat Detection Techniques," Proc. 11th Annu. Cyber Inf. Secur. Res. Conf., Apr. 2016, doi: 10.1145/2897795.2897799.

[4] "Insider Threat Test Dataset." https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=508099 (accessed Feb. 13, 2023).

[5] L. Liu, O. De Vel, C. Chen, J. Zhang, and Y. Xiang, "Anomaly-Based Insider Threat Detection Using Deep Autoencoders," 2018 IEEE Int. Conf. Data Min. Work., vol. 2018-November, pp. 39–48, Feb. 2018, doi: 10.1109/ICDMW.2018.00014.

[6] N. Bhodia, P. Prajapati, F. Di Troia, and M. Stamp, "Transfer Learning for Image-Based Malware Classification," ICISSP 2019 - Proc. 5th Int. Conf. Inf. Syst. Secur. Priv., pp. 719–726, 2019, doi: 10.5220/0007701407190726.

[7] G. R. G, A. Sajjanhar, and Y. Xiang, "Image-Based Feature Representation for Insider Threat Classification," Appl. Sci., vol. 10, no. 14, Nov. 2019, doi: 10.3390/app10144945.

[8] A. Sharma, E. Vans, D. Shigemizu, K. A. Boroevich, and T. Tsunoda, "DeepInsight: A methodology to transform a non-image data to an image for convolution neural network architecture," Sci. Reports 2019 91, vol. 9, no. 1, pp. 1–7, Aug. 2019, doi: 10.1038/s41598-019-47765-6.