Unsupervised User-Based Insider Threat Detection Using Bayesian Gaussian Mixture Models

Simon Bertrand  
Informatique et génie logiciel  
Laval University  
Québec, Canada  
siber93@ulaval.ca

Josée Desharnais  
Informatique et génie logiciel  
Laval University  
Québec, Canada  
josee.desharnais@ift.ulaval.ca

Nadia Tawbi  
Informatique et génie logiciel  
Laval University  
Québec, Canada  
nadia.tawbi@ift.ulaval.ca

Abstract—Insider threats are a growing concern for organizations due to the amount of damage that their members can inflict by combining their privileged access and domain knowledge. Nonetheless, the detection of such threats is challenging, precisely because of the ability of the authorized personnel to easily conduct malicious actions and because of the immense size and diversity of audit data produced by organizations in which the few malicious footprints are hidden. In this paper, we propose an unsupervised insider threat detection system based on audit data using Bayesian Gaussian Mixture Models. The proposed approach leverages a user-based model to optimize specific behaviors modelization and an automatic feature extraction system based on Word2Vec for ease of use in a real-life scenario. The solution distinguishes itself by not requiring data balancing nor to be trained only on normal instances, and by its little domain knowledge required to implement. Still, results indicate that the proposed method competes with state-of-the-art approaches that use stronger hypotheses, presenting a good recall of 88%, accuracy and true negative rate of 93%, and a false positive rate of 6.9%. For our experiments, we used the benchmark dataset CERT version 4.2.

Index Terms—Insider Threat, Bayesian Gaussian Mixture Model, Gaussian Mixture Model, Unsupervised learning, Word2Vec

I. INTRODUCTION

Insider threats occur when a privileged member of an organization wrongfully uses their access in a way that causes harm to their organization. Those actions can be intentional, as in the case of theft or sabotage, or not. The unintentionally dangerous insider is mostly acting by negligence or misinformation. An example is an employee who copies corporate sensitive data locally for convenience, thus creating a second access point to the information which can then be exploited by a hacker. Hence, the insider threat is a broad type of cyber menace which makes its detection particularly difficult.

Confidentiality, integrity, and availability of information are increasing concerns for organizations. Even though insider threats are only a fraction of all existing cyber threats, this type of menace presents a real and unique danger for organizations. Firstly, an insider threat can be more damaging to an organization than a traditional cyberattack. This is mainly explicable by the privileged accesses and great domain knowledge that the insider possesses over an outsider. The insider has then a better opportunity to carry out efficiently and quietly the attack.

Insider threat detection has attracted many researchers in the last decades. One common strategy is by modeling the behaviors of the users and identifying any significant divergence as a potential threat. In that matter, audit data, describing the activity of every member of an organization in the network, are regularly chosen to learn user behaviors using statistical or machine learning models.

However, detecting insider threats based on audit data presents many challenges, one of which is to efficiently consider sequenced-based behavior features. Indeed, like most cyber threats, an insider attack is rarely defined by a single malicious event, but mostly as a series of events. Additionally, not only can the malicious series of events be scattered over a period of time, but they are also often sequence dependent. For instance, considering a simple data exfiltration threat, the event of reading sensitive data before writing an email is more suspicious than the other way around. Few machine learning-based works focus on using the event sequence information in the behavior learning process on long time windows.

Another challenge is to create a solution that is flexible and adapted for real-life organizations. The singularity and complexity of all organizations’ technology architecture are hard to reproduce in public datasets. While using public datasets is convenient for comparison purposes, one needs to be careful. One risk is to overfit a solution to a specific dataset, leading to poor performance in other settings. For instance, using the label information in the dataset can be convenient to balance the data classes or to extract positive instances for one-class training. However, organizations rarely possess historically labeled audit data, which prevents them to train the proposed models on their specific audit data. In such a case, only a pre-trained model can be used, which requires high similarity between the organization’s technological architecture and the dataset’s, which is unlikely. Furthermore, in some cases, organizations can even hardly guarantee that historical audit data are threat free, which limits the use of One-Class models. Those limitations make supervised solutions unsuitable for organizations and highlight a need for unsupervised alternatives.
A. Contributions

Considering those challenges, we propose an insider threat detection system that uses unsupervised machine learning trained on processed audit data. More precisely, the technique consists of training a Bayesian Gaussian Mixture Model (BGMM) for every user, utilizing their historical audit data to learn normal behavior clusters/components.

To address the challenge of sequence dependencies of insider threats, and to facilitate feature extraction, a user-based Word2Vec model is trained to generate a daily activity summary vector, capturing contextual information about the activities in the host logs.

We also propose the use of a custom number of clusters/components for every user on insider threat detection performance, which will later be compared with a fix number of components. There is, to our knowledge, no existing work that combines an automatic number of clusters/components with user-based models for insider threat detection. Furthermore, few insider threat detection approaches deal with using events sequential information on long time windows, as the proposed daily feature vectors. Thus, we believe that the solution is novel.

The proposed method is developed with the restriction of requiring as little domain knowledge and data pre-processing as possible, thus increasing its flexibility. In this regard, our solution does not require data balancing nor to be trained only on normal instances. Yet the proposed approach outperforms state-of-the-art methods in the accuracy, false positive rate, and true negative rate metrics, and still offers a competitive recall rate. In addition, the simplicity of the feature extraction process, for the daily activity summary vectors, also contributes to making the solution more suitable for a real-life scenario.

B. User-based Models

Most papers in the field propose coarser models than the one we chose, like role-based or organization-based models. We regroup in this section the benefits and drawbacks of using coarser models and motivate our choice of using user-based models.

Even though coarser models offer better user behavior comparison, because each model is trained using the data of a group of users, the relevance of such models depends entirely on proper user grouping. Indeed, irrelevant user grouping could lead to naturally different behaviors being compared together and thus could force the model to overgeneralize.

For instance, one could argue that role-based models mitigate the risk of grouping incomparable users because of the obvious correlation between a user’s work activities and his role. However, even if in some fields, like healthcare, the employees’ roles are well defined, this is not always the case. For instance, information technology employees are often assigned a set of responsibilities instead of a specific role. Thus, role-based models in real organizations can be challenging by requiring considerable efforts to efficiently group comparable users to ensure the relevance of the models.

Most papers avoid entirely the need to find relevant user groupings by modeling an organization-based model. Such solutions use one model that is trained on all users’ audit data regardless of their role in the organization. Even though organization-based models offer a simple approach to insider threat detection that considers user behavior comparison, such models can suffer from overgeneralization and miss out on more precise user behavior variations. Indeed, by training on the behaviors of all users, an organization-based model could be more at risk of missing abnormal behaviors for a user if the same behavior is normal and frequent for other users. For instance, a nurse who accesses financial data, which is normal behavior for accountants, but questionable for a nurse. Thus one could question the relevance of using organization-based models in big organizations having employees working in multiple fields.

A considerable alternative is a user-based model that does not require any user grouping efforts and allows more specific behavior learning. Indeed, our intuition is that training only on one user’s data improves the model’s precision regarding what are normal behaviors for a specific user. Hence, such models should flag any significant change in behavior from what was observed for a user regardless of his role. Furthermore, we believe that user-based models offer opportunities for custom model optimizations depending on each user. For instance, in experiments, we observed that selecting a custom number of components, through BGMMs, on user-based models provides better results than what is presented in state-of-the-art techniques. Further optimizations could be of interest for future work like the optimization of BGMM-specific parameters.

Thus, our choice to develop a user-based insider threat detection system is motivated by the fact that no user grouping efforts are necessary and that such models should be more precise regarding user behavior variations. Moreover, our choice is motivated by the many optimization opportunities that user-based models offer.

In addition, because few papers explore user-based solutions for insider threat detection, we believe that the proposed solution is novel in the field and could arouse interest in such modelization for future research.

However, a challenge of using user-based models is ensuring that the user data is sufficient and mostly benevolent for learning the behaviors. Therefore, such modelization will perform poorly on new users whose intentions are malicious from the start. Even though those challenges should be studied further in future work, in this paper we assume that user data is sufficient and mostly benevolent. Observe that our method does not depend crucially on the granularity of the analysis. Complete anomaly detection should include both types of analysis from time to time.

II. RELATED WORK

Previous work in unsupervised insider threat detection based on audit data can principally be grouped into two categories:
signature-based and machine learning-based techniques.

Signature-based threat detection techniques mainly consist of creating a dictionary containing allowed and/or unallowed activity patterns. The dictionary is then prompted to check if any sequence of activity matches those in the lexicon to determine if the behavior is normal or abnormal. Signature-based techniques offer the advantage of a low false positive rate but require frequent updates to detect new anomalies [1] which in turn require frequent updates of the dictionary. In other words, an important limitation of signature-based techniques is their inability to detect unknown threats.

Machine learning techniques present an appealing alternative to signature-based methods because of their ability to automatically learn normal and abnormal behaviors. This generally increases their flexibility and reduces the required domain knowledge [2]. In the last decades, many insider threat detection systems using unsupervised machine learning were constructed on statistical or clustering techniques. For instance, Eldardiry et al. [3] propose a user-based model system that uses the K-Means algorithm to model daily user behavior. Happa & Tabash [4] present a similar framework but Gaussian Mixture models (GMMs) are used to detect anomalous instances by selecting the data points that are less likely to have been generated by the learned Gaussian distributions. Kim et al. [5] propose a study on the existing clustering and statistical techniques presenting the performance of K-Means, Parzen Window Density Estimation, Principal Component Analysis, and Gaussian algorithms. In general, statistical and clustering techniques offer a simple way to detect the malicious insider, but suffer from a high false positive rate [6].

A leap in the field of unsupervised insider threat detection has been the use of Autoencoders. Zhang et al. [7] propose the use of compressed daily feature representations obtained with a Denoising Autoencoder as input to a GMM, to learn normal behaviors and detect divergent instances. The solution stands out from other techniques partly because of the use of a Word2Vec model for automatic feature extraction, which inspired part of this work.

With the increasing popularity of deep learning in recent years, many researchers integrated the use of deep models into their approach to detecting insider threat. Many RNN-based methods have been proposed to detect insider threats because of their ability to model time series data. However, the conventional RNN tends to perform poorly in long time series [8], which can be problematic while using audit data, which are usually quite dense, for anomaly detection tasks. This limitation is the reason why most current efforts in the deep learning field use LSTM neural networks which are known to capture long-time dependencies [9]. As examples, Nasir et al. [10] and Sharma et al. [11] propose the use of LSTM-AutoEncoders to encode and decode session activity summary vectors and label the sessions with the highest reconstruction error score as being of malicious nature. Results demonstrate that deep learning methods provide overall better results compared to shallow machine learning models, specifically a lower false positive rate.

Finally, even though we focus on insider threat detection, we highlight some of the recent work in general anomaly detection. For instance, Meijaard et al. propose an interesting review of the use of the Isolation Forest algorithm for the detection of International Revenue Sharing Fraud (IRSF) on Voice over IP communications [12]. Furthermore, there is recent progress in anomaly detection from images, for instance, Zolfaghari et al. propose an unsupervised anomaly detection system based on convolutional neural networks to detect peculiarity in images [13]. In the same field, Wang et al. propose an anomaly detection model based on video surveillance footage and deep learning [14]. However, even if detecting anomaly seems to be the same goal that we pursue, images are very different objects than tabular sequential data, which we are interested in. In other words, even if the preceding initiatives are novel, we believe there is no way to compare results between these fields and ours.

III. METHODOLOGY

Our solution consists in learning the daily behaviors of every user in an organization to then detect any day diverging significantly from the typical user behavior as being anomalous. In this work, as for most approaches in this field, What defines an anomalous instance is the presence of at least 1 malicious event in the chosen time window, being the day in our case. To do so, we use a feature extraction process that necessitates as little domain knowledge as possible and user-specific behavior learning models.

The proposed framework is presented in Fig. 1. It is separated into a data pre-processing phase and a behavior learning/insider detection phase that we explain in the following paragraphs.

A. The dataset CERT 4.2

The dataset used in this study is the commonly used CERT insider threat dataset version 4.2 [15] created by Carnegie Mellon’s Software Engineering Institute [16]. This synthetic dataset is composed of audit data generated by simulating a 1000 employees organization within 502 days. The audit data is separated into 5 domain types: logon, emails, files, HTTP, and devices. Each domain contains one or two specific activity types. There are 32,770,227 events in total. Table I presents the different domains and activity types. Each event from a specific domain is stored in a CSV file. The dataset is mainly composed of normal events, but malicious activities perpetrated by 70 users were injected and account for only 0.03% of the dataset’s daily instances. Consequently, one challenge regarding the use of this dataset is that the normal and malicious instances are imbalanced. Even though balancing efforts have been proven, in other work, to be beneficial for the detection of malicious insiders [17], in this work no data balancing is performed. This decision is motivated by our goal to create a solution that is more appropriate for a real-world scenario, where it is difficult to have any knowledge about past anomalous actions and where it is fair to assume that the data is also unbalanced.
TABLE 1
DATASET ACTIVITIES DESCRIPTION

| Domain | Activity | Description       |
|--------|----------|-------------------|
| Logon  | Logon    | Connection using the user-id |
| Logon  | Logoff   | Disconnection of the user-id |
| HTTP   | HTTP     | Website access     |
| Email  | Email    | Creation of an email |
| File   | File     | File-level access  |
| Device | Connect  | Connection of a USB device |
| Device | Disconnect | Disconnection of a USB device |

B. Data Pre-processing

1) User-based Data Extraction: Using the CERT dataset, the activity in the 5 domains is first aggregated for every single user. Furthermore, in a user-based model configuration, this data grouping is essential.

The user’s audit data are finally grouped by day, so every instance represents the daily activities of an individual, for a total of 330,452 instances, with only 966 having at least a malicious activity occurring during the day. In this work, we only consider the activity type itself and the order in which it occurs to create the behavioral model, so every other information is ignored. Motivated by our objective to create an easy-to-implement solution, the choice to keep only the activity type is driven by its standardness and the little domain knowledge required to identify it. In summary, the results of this step are string objects composed of event types carried out by a user daily.

2) Feature Extraction: One of the difficulties in generating feature vectors representing a condensed summary of a time window is capturing the temporality and dependencies of events. In that matter, we use Word2Vec models [18]. Using the Skip-gram model, Word2Vec can capture syntactic and semantic word relationships automatically [19], which allows, in our case, to capture the user’s daily specific behavior patterns like the typical order of the activities. Specifically, a Word2Vec model is trained for every user, using their daily activity strings. This step ensures that every user has custom word embeddings, depending on their behaviors. Word2Vec models contribute to our goal of reducing the dependency of the domain knowledge [7], by automating the extraction of complex features from simple data, in our case the activity type. There exist other popular models to generate word embeddings like FastText [20], that could be explored in future work.

Finally, using the trained Word2Vec models, every user’s daily activities are summarized into a vector for every day they were active in the organization. To do so, the Word2Vec model receives one by one the daily activities of a user and transforms every single activity into an embedding that is then summed with the other activities embedding that occurred during that same day. The resulting vector is then a daily summary that has information about the volume and type of events carried out and contains contextual details about the order of execution of the user’s daily activities. The proposed feature extraction process is illustrated in Figure 2.

Note that the embeddings are generated from past data hence it means that if used in a real-time scenario, the new sequences will not have an impact on the embeddings generated.

C. Behavior Learning

In the pre-processing phase, activity vectors, each summarizing a user’s daily events, are extracted for every user. To learn the user’s normal behaviors using those vectors, we opted for Gaussian Mixture Models because of their popularity in the field, which can be explained by their high recall rate [7]. Thus, we assume that user data follow multiple Gaussian distributions. Choosing a multimodal distribution is even more relevant when considering the fact that employees can exhibit more than one behavior. Furthermore, assuming that a user can have more than one normal behavior, we rely on the GMM’s ability to learn from multimodal data distributions.
Fig. 2. The Feature Extraction Process

to fit clusters/components representing behaviors. Moreover, because of the low volume of malicious records in the dataset, our intuition is that every component will mostly describe a normal behavior for the user, and thus high-density regions in a component should describe normal behaviors, and low-density ones should describe abnormal behaviors. In other words, we deal with the detection of abnormal behaviors as an outlier detection task.

However, GMMs present the challenge of selecting a relevant number of components. For that reason, we are interested in the effects of an automatic solution for the selection of the number of components in a user-based framework. In that regard, we leverage the fact that the solution is user-based, to optimize the performance by selecting an efficient number of components for every user as opposed to using the same number of components for all users. Because every user has unique behaviors, using a custom number of components depending on each user should improve modelization.

A simple example, to explain further this intuition, is a user that exhibits two normal behaviors, the first one being their email-intensive days, maybe occurring early on in the week and the second one being characterized by higher file and website activities. For this particular user, a 2 component model would probably suffice. A higher number of components, in this case, could mean overfitting the data. In an anomaly detection task using partitioning models, overfitting a model can lead to the creation of a separate cluster, grouping the anomalies and making them appear normal by most distance or density metrics. However, for another user having a specific behavior depending on the day of the working week, a 5 component model could be more appropriate. In that last example, choosing a lower number of components could mean underfitting the data, and thus overgeneralizing the behaviors.

Many options are available to optimize the number of components of GMMs, like the Bayesian Information Criterion (BIC) score and the Akaike Information Criterion (AIC) score, but in this work we select a variation of the GMM: the Bayesian Gaussian Mixture Model which is a GMM but using the Dirichlet process to infer a weight of importance for every component. This technique only requires that a superior number of components be provided as input.

Prior to the Bayesian Gaussian Mixture Model, we present the Gaussian Mixture Model.

1) The Gaussian Mixture Model (GMM): The GMM is a probabilistic model often used in anomaly detection tasks, due to its high recall rate [7]. In this model, it is assumed that the data obey a mixture of several Gaussian distributions defined as:

$$
N(x|\mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^K |\Sigma_i|}} e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)},
$$

(1)

where parameters are such as described in Table II. Such a mixture of distributions could appear for instance when analyzing a user’s daily activity summaries: a Gaussian distribution could explain email-intensive days and another Gaussian distribution could explain file-intensive days.

Knowing the number of components, the GMM learns to fit the instances in a way that maximizes the log-likelihood of the dataset, defined as:

$$
\log p(x|\mu, \phi, \Sigma) = \sum_{n=1}^{N} \log \sum_{i=1}^{K} \phi_i N(x|\mu_i, \Sigma_i).
$$

(2)

To do so, the Expectation Maximization algorithm [21] learns the \( \mu, \phi \) and \( \Sigma \) parameters that maximize Equation (2) for the dataset.

The probability of a single instance generated by the mixture model is calculated as:

$$
p(x) = \sum_{i=1}^{K} \phi_i N(x|\mu_i, \Sigma_i),
$$

(3)
which is simply the sum of the probability of the instance to be part of each Gaussian distribution multiplied by the prior probability/weight of the component.

Even though GMMs are often used in anomaly detection tasks, the choice of a good number of components can be a difficult chore. The optimal number of components will vary from user to user because the user’s behaviors are unique. Therefore, a potential gain can be achieved by having a per-user model for which a custom number of components is selected.

2) The Bayesian Gaussian Mixture Model (BGMM): The BGMM is very similar to a GMM except for the fact that it is a non-parametric model, meaning it uses variational inference to estimate the model parameters. Precisely, the BGMM uses prior distributions over the parameters of the GMM to infer an efficient number of components from the data. Then the parameters optimization process follows the Expectation Maximization algorithm but computes the entire posterior distribution over the parameters for regularization.

For instance, the proposed BGMM uses the same parameters as described in Table II, but with a Dirichlet process as a distribution to infer the number of components $\phi$.

D. Insider Threat Detection

During the learning process, a score for every user’s day is calculated using the trained BGMM. The score represents the log-likelihood of the instance compared to its model, which is the log of Equation (3). A low score means that the day is at the further end of our learned Gaussian distributions or and is part of a lower-weight component. Thus, we interpret a low score as potential abnormal behavior. Every day’s score is divided by the mean of the user’s scores to get a ratio of that day’s score against what is normal for the user. This is motivated by the fact that what defines a normal score varies from one user to another, and so using this ratio is more relevant to compare across users. Finally, with the daily score ratios of every user, every day that has a greater score than a threshold is identified as malicious. Note that by using a daily anomaly score, the model’s predictions are not interpretable, which should be an improvement objective for future work.

The proposed method is based on transductive learning, meaning that all available instances are used to train the models and to infer their anomaly score (log-likelihood). This implies that the solution is designed to catch past threat in an organization. Thus, the proposed approach is best suited for organizations that have no prior knowledge about the normality of their historical audit logs. In that case, the proposed solution allows an organization to gain information about past anomalous behaviors, thus allowing it to improve its security policies and implement efficient preventive measures with the objective of limiting its exposure to future attacks.

Even though nothing prevents an organization to use the proposed approach on future/unseen data, greater care must be taken. Indeed, because it would be unrealistic to retrain the entire user model for each new workday, one could only calculate the log-likelihood of the new instance using the trained model. However, doing so increases the importance of ensuring that past user data is sufficient to avoid non-representative models being used as the user behaviors baseline, resulting in a higher risk of falsely labeling new data as anomalous. In that case, the longer the weak models are used as baselines for user behaviors without retraining, the greater the negative impact on the quality of the solution’s overall performance.

Although the suggested use case of the solution could also be impacted by insufficient user data, we believe that the effects on the overall performance are mitigated, precisely, because of the small amount of data concerned. In other words, assuming that the majority of an organization’s employees present sufficient user data, only a minority of users will have a less reliable behaviors model due to scarce audit logs. Furthermore, considering that this minority of weaker models only make predictions on the data available for the corresponding user, which is scarce, we believe that the effect on the solution’s overall performance is negligible. This is supported by the results presented further. Nevertheless, future work should study the impact of using few data for training user behaviors models and improve the proposed solution in this regard.

IV. IMPLEMENTATION AND RESULTS

In this section, we evaluate the performance of our solution. We first compare it with similar state-of-the-art techniques. Then, we verify our assumption that a custom number of components for every user is beneficial. Finally, we check if the performance varies between executions.

The implementation is done using an Ubuntu 20.04.4 LTS operating system. The language used is Python. Scikit-learn’s [22] Bayesian Gaussian Mixture Model and Gensim’s Word2Vec libraries are used. For the following experiments, the vector size of the Word2Vec word embeddings is set to 10, with two more configurations presented later on and in Table VI. The maximum number of components for the BGMM models is set to 5, which generally corresponds to the maximum we saw in other clustering-based insider threat detection research. All of the data is used, for training and detection in an unsupervised way. The proposed method does not require data balancing or to be trained only on normal instances.

To evaluate our solution we use the false positive rate (FPR), recall, true negative rate (TNR), and accuracy metrics, being common metrics to evaluate anomaly detection models. The metrics formulae are as follows, where a negative instance refers to a normal day and a positive to a day containing at least one malicious event:

$$\text{FPR} = \frac{\text{False Positives}}{\text{True Negatives} + \text{False Positives}}$$

$$\text{TNR} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}}$$

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Instances}}$$
\[
FPR = \frac{FP}{TN + FP}
\]
\[
Recall = \frac{TP}{TP + FN}
\]
\[
TNR = \frac{TN}{TN + FP}
\]
\[
Accuracy = \frac{TP + TN}{TP + TN + FP + FN},
\]

where \(TN\) is true negative, \(FP\) is false positive, \(TP\) is true positive and \(FN\) is false negative.

Table III presents a comparison with other methods. We include the most recent research, even if old, using the CERT Version 4.2 dataset based on unsupervised machine learning techniques. Based on those criteria, we selected, to the best of our knowledge, state-of-the-art papers using popular machine learning techniques and with which a comparison with our technique is relevant. They all use an organization-based model. Except for the models used, what mostly differentiate the solutions is the feature extraction process. Indeed, Sharma et al. [11] and Lin et al. [23] select manually relevant features while Zhang et al. [7] and the proposed method rely on automatic features extraction. The selected metrics for each method are displayed if they are available in the corresponding paper. We could not confidently calculate the missing metrics from the concerned papers. The results of our approach were obtained by calculating the mean of 100 random executions, for validity.

Table III shows that the proposed approach outperforms other comparative techniques on the accuracy, false positive rate, and true negative rate metrics and still presents a competitive recall. Furthermore, it is important to mention that the presented results were achieved even if the proposed method does not rely upon more domain knowledge-intensive attributes, data balancing, or one-class training, as opposed to other techniques.

Assuming that the selected dataset is representative of real user behaviors, the results obtained suggest that user-based models can be relevant to detect insider threats in an organization, even if the amount of user data varies.

We have included the accuracy metric in the table, but we believe it can be misleading because the dataset is imbalanced. For instance, in such imbalanced datasets, the best accuracy is often obtained by predicting the majority class to all examples. Consequently, the F1-score is often preferred. It is calculated using precision and recall or, equivalently, as follows:

\[
F1\text{-score} = \frac{TP}{TP + 0.5 \times (FP + FN)}
\]

F1-score is preferred in general but it is rarely presented in papers on unsupervised insider threat detection. We obtain an F1-score of 7%, which is low, of course, but it is as expected.

Table IV presents the confusion matrix of one execution of the solution. A strong diagonal can be observed, with 860 days containing malicious activities accurately detected out of 966 in total (for a recall of 89% for this particular execution). Most normal days are correctly identified, with 306 526 days accurately predicted out of 329 486. Even though 22 960 days were wrongfully labeled as anomalous, which is quite high, it is a common problem with unsupervised anomaly detection methods. Nonetheless, a false positive rate of around 7% still outperforms similar state-of-the-art techniques. Furthermore, one could argue that in this particular field, a higher false positive rate is better than missing out on falsely labeled benign attacks. However, security analysts should always act with caution upon treating any threat prediction to ensure its legitimacy and avoid any false allegation against benevolent employees.

We present the receiver operating characteristic or ROC curve to evaluate the performance of our binary classifier. This curve shows the recalls, on the y-axis, and false positive rates, on the x-axis, obtained at different threshold configurations. This representation can be a good way for cyber analysts with different time budgets to find a good compromise by choosing a threshold that does not lead to an unrealistic amount of false positives and offers an expected recall rate they are comfortable with. Because it can be hard to draw valid conclusions from a ROC curve, the area under the ROC curve (AUC) is a metric to evaluate and compare a model’s performance. Figure 3 presents the ROC curve of a random instance of the proposed method. The curve shows that the proposed solution offers many interesting opportunities for insider threat detection depending on an organization’s available resources. The proposed solution presents an AUC score of 0.958. To compare with another state-of-the-art method using the same dataset, an AUC score of 0.949 is achieved by Sharma et al. [11].

Figure 4 presents the recall of 100 random executions of the solution. In other words, each point in the graph presents the recall of one execution of our algorithm. It shows that the performance does not vary greatly and so is not substantially affected by randomness.

We also confirm our intuition that performance could be optimized by a per-user custom number of components. To do so, we compare our results with three configurations of the traditional GMM in Table V. All the pre-processing is the same as the proposed solution, only the model itself is changed for a GMM. Even the score function is identical because, as seen earlier, the GMM and the BGMM are very similar and only differ in their parameter learning steps. The three configurations are GMMs for every user with 1, 3, and 5 components. Results suggest that a custom number of components for every user is beneficial, beating every other fixed number of component configurations in every metric.

Finally, Table VI presents the proposed solution performance depending on the embedding size of the events, which also determines the daily embedding size. The embedding size has an impact on performance. This is mainly noticeable with the smaller embedding size setting of five that performs poorly. This could be explained by the complexity of encoding the information about event sequences in a small vector.
### TABLE III
#### RESULTS COMPARISON

| Method               | Reference | Recall  | FPR    | Accuracy | TNR    |
|----------------------|-----------|---------|--------|----------|--------|
| LSTM-AutoEncoder     | [11]      | 91.03%  | 9.84%  | 90.17%   | 90.15% |
| DBN-OCSVM            | [23]      | 81.04%  | 12.18% | 87.79%   | NA     |
| DA With Clustering   | [7]       | 88.9%   | 20%    | 75%      | NA     |
| Proposed             | NA        | 88.38%  | 6.9%   | 93.08%   | 93.10% |

Fig. 3. ROC Curve

### TABLE IV
#### CONFUSION MATRIX

|                  | Insider | Normal |
|------------------|---------|--------|
| **Predicted Class** |         |        |
| Insider           | 863     | 106    |
| Normal            | 22,960  | 306,526|

### TABLE V
#### FIXED VERSUS CUSTOM NUMBER OF COMPONENTS

| METHOD   | Recall | FPR    | Accuracy | TNR    |
|----------|--------|--------|----------|--------|
| GMM-1    | 78.05% | 9.02%  | 90.94%   | 90.98% |
| GMM-3    | 78.88% | 8%     | 91.95%   | 92%    |
| GMM-5    | 78.05% | 7.28%  | 92.68%   | 92.71% |
| BGMM     | **88.38**% | 6.9%   | **93.08%** | **93.10%** |
V. Conclusion

In this paper, an unsupervised insider threat detection model is proposed and tested on the CERT dataset. The dataset is grouped by user and day, and only the ordered activity type is kept. Word2Vec model is used to generate user-specific activity embeddings, capturing activity sequence information. BGMMs are finally used with the daily summary vectors to train and detect malicious behaviors. The performance of the proposed method is competitive with state-of-the-art techniques in addition to having the advantages of not requiring data balancing nor being trained only on normal data, all of which with minimal domain knowledge required. Therefore, the proposed solution not only distinguishes itself by its performance but also by its flexibility and feasible nature. Even though the proposed method performs well using only the activity type and its order, we believe that further improvements could be made by integrating other relevant information like the time and content of the activity and the weekday. The addition of a relevant way to integrate comparison of user behaviors could also be an interesting improvement for the proposed solution. In future work, we would like to improve the daily embedding generation process to make it more suitable for real-time execution and explore other embedding models. The needed time consumption of such a solution should also be studied. Additionally, the scenario of the mostly malicious user and the impact of insufficient user data should also be considered even if on the selected dataset results obtained suggest that the solution is relevant regardless of those possibilities. Finally, we also want to explore the BGMM’s performance in a real-time setting, and precisely study the inference of the number of components in an evolving environment.

REFERENCES

[1] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, “Survey of intrusion detection systems: techniques, datasets and challenges,” Cybersecur., vol. 2, p. 20, 2019.
[2] L. Liu, C. Chen, J. Zhang, O. De Vel, and Y. Xiang, “Unsupervised insider detection through neural feature learning and model optimisation,” in Network and System Security, J. K. Liu and X. Huang, Eds. Cham: Springer International Publishing, 2019, pp. 18–36.
[3] H. Eldardiry, E. Bart, J. Liu, J. Hanley, B. Price, and O. Brdiczka, “Multi-domain information fusion for insider threat detection,” in 2013 IEEE Security and Privacy Workshops, 2013, pp. 45–51.
[4] K. Al Tabash and J. Happa, “Insider-threat detection using gaussian mixture models and sensitivity profiles,” Computers & Security, vol. 77, pp. 172–180, 2020.

[5] J. Kim, M. Park, H. Kim, S. Cho, and P. Kang, “Insider threat detection based on user behavior modeling and anomaly detection algorithms,” Applied Sciences, vol. 9, no. 19, 2019. [Online]. Available: https://www.mdpi.com/2076-3417/9/19/4018

[6] Y. Chen, S. Nyemba, and B. Malin, “Detecting anomalous insiders in collaborative information systems,” IEEE Transactions on Dependable and Secure Computing, vol. 9, no. 3, pp. 332–344, 2012.

[7] Z. Zhang, S. Wang, and G. Lu, An Internal Threat Detection Model Based on Denoising Autoencoders, 01 2020, pp. 391–400.

[8] S. Althubiti, W. Nick, J. Mason, X. Yuan, and A. Esterline, “Applying long short-term memory recurrent neural network for intrusion detection,” in SoutheastCon 2018, 2018, pp. 1–5.

[9] Y. Fu, F. Lou, F. Meng, Z. Tian, H. Zhang, and F. Jiang, “An intelligent network attack detection method based on rnn,” in 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC), 2018, pp. 483–489.

[10] R. Nasir, M. Afzal, R. Latif, and W. Iqabl, “Behavioral based insider threat detection using deep learning,” IEEE Access, vol. PP, pp. 1–1, 10 2021.

[11] B. Sharma, P. Pokharel, and B. Joshi, “User behavior analytics for anomaly detection using lstm autoencoder - insider threat detection,” in Proceedings of the 11th International Conference on Advances in Information Technology, ser. IAIT2020. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3406601.3406610

[12] Y. J. Meijaard, B. C. M. Cappers, J. G. M. Mengerink, and N. Zannone, “Predictive analytics to prevent voice over ip international revenue sharing fraud,” in Data and Applications Security and Privacy XXXIV, A. Singhal and J. Vaidya, Eds. Cham: Springer International Publishing, 2020, pp. 241–260.

[13] M. Zolfaghari and H. Sajedi, “Unsupervised anomaly detection with an enhanced teacher for student-teacher feature pyramid matching,” in 2022 27th International Computer Conference, Computer Society of Iran (CSICC), 2022, pp. 1–4.

[14] L. Wang, H. Tan, F. Zhou, W. Zuo, and P. Sun, “Unsupervised anomaly video detection via a double-flow convlstm variational autoencoder,” IEEE Access, vol. 10, pp. 44 278–44 289, 2022.

[15] “Insider threat test dataset. (2016),” https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=508099, retrieved January 2022.

[16] J. Glasser and B. Lindauer, “Bridging the gap: A pragmatic approach to generating insider threat data,” in 2013 IEEE Security and Privacy Workshops, 2013, pp. 98–104.

[17] D. C. Le, N. Zincir-Heywood, and M. I. Heywood, “Analyzing data granularity levels for insider threat detection using machine learning,” IEEE Transactions on Network and Service Management, vol. 17, no. 1, pp. 30–44, 2020.

[18] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” in 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2–4, 2013, Workshop Track Proceedings, Y. Bengio and Y. LeCun, Eds., 2013. [Online]. Available: http://arxiv.org/abs/1301.3781

[19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in Advances in Neural Information Processing Systems, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds., vol. 26. Curran Associates, Inc., 2013.

[20] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information,” Trans. Assoc. Comput. Linguistics, vol. 5, pp. 135–146, 2017. [Online]. Available: https://doi.org/10.1162/tacl_a_00051

[21] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” Journal of the Royal Statistical Society. Series B (Methodological), vol. 39, no. 1, pp. 1–38, 1977. [Online]. Available: http://www.jstor.org/stable/2984875

[22] L. Buinick, G. Loupe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, “API design for machine learning software: experiences from the scikit-learn project,” in ECML PKDD Workshop: Languages for Data Mining and Machine Learning, 2013, pp. 108–122.