Robust Fraud Detection via Supervised Contrastive Learning

Vinay M.S.
vmadanbh@uark.edu
University of Arkansas
Fayetteville, Arkansas, USA

Shuhan Yuan
Shuhan.Yuan@usu.edu
Utah State University
Logan, Utah, USA

Xintao Wu
xintaowu@uark.edu
University of Arkansas
Fayetteville, Arkansas, USA

ABSTRACT

Deep learning models have recently become popular for detecting malicious user activity sessions in computing platforms. In many real-world scenarios, only a few labeled malicious and a large amount of normal sessions are available. These few labeled malicious sessions usually do not cover the entire diversity of all possible malicious sessions. In many scenarios, possible malicious sessions can be highly diverse. As a consequence, learned session representations of deep learning models can become ineffective in achieving a good generalization performance for unseen malicious sessions. To tackle this open-set fraud detection challenge, we propose a robust supervised contrastive learning based framework called *ConRo*, which specifically operates in the scenario where only a few malicious sessions having limited diversity is available. ConRo applies an effective data augmentation strategy to generate diverse potential malicious sessions. By employing these generated and available training set sessions, ConRo derives separable representations w.r.t open-set fraud detection task by leveraging supervised contrastive learning. We empirically evaluate our ConRo framework and other state-of-the-art baselines on benchmark datasets. Our ConRo framework demonstrates noticeable performance improvement over state-of-the-art baselines.

KEYWORDS

fraud detection, contrastive learning, open-set, augmentation

ACM Reference Format:
Vinay M.S., Shuhan Yuan, and Xintao Wu. 2018. Robust Fraud Detection via Supervised Contrastive Learning. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Computing platforms, such as social networking sites and cloud systems, experience large volumes of malicious or fraudulent activities due to the anonymity and openness of the Internet. It is critical to identify such malicious activities in order to protect legitimate users. In practice, the activities of a user are usually modeled as an activity session. For example, in a computer system, an activity session is a sequence of user activities starting with log-in and ending with log-out. A popular approach for detecting malicious sessions is through deep learning models [15]. The main idea is to derive session representations by making normal sessions deviate from malicious ones in the representation space for deriving anomaly scores.

In many real-world fraud detection scenarios, only a few labeled malicious and an abundance of normal sessions are available [15, 16]. These few available malicious sessions usually do not sufficiently cover the entire diversity of all possible malicious sessions. It is well known that malicious sessions can be highly diverse [15]. Many attackers keep evolving their activity patterns to avoid detection. Such malicious sessions are usually not available for training a deep learning model. Suppose a deep learning model is trained by utilizing a few available malicious sessions. Now in the testing phase, due to the large diversity in the possible malicious sessions, the test set distribution might be different from the training set distribution. For example, the training set might contain only a few types of malicious sessions, and the test set might include other types of malicious sessions that are not observed in the training set. Hence, the learned session representations by using these few malicious sessions in the training set might not be discriminative enough to achieve good generalization on detecting unseen malicious sessions. Clearly, the fraud detection task is essentially an *open-set* detection task.

The existing deep anomaly detection approaches which operate on the setting of a few available anomalous samples, employ metric learning [12, 13] or deviation loss based learning [9, 10]. These approaches attempt to obtain a decision boundary by using a few available anomalies. However, these approaches can easily overfit w.r.t seen anomalies and can suffer from poor generalization performance if the anomalies encountered during the testing stage deviate from the training set anomalies [1]. To address this challenge, recently Ding et al. [1] presented a novel open-set deep anomaly detection approach. They train their model to detect unseen anomalies by jointly employing: (1) a data augmentation strategy through which they generate augmented samples that can closely resemble unseen anomalies, and (2) learning in the latent residual representation space. However, their approach has been specifically designed to operate on image data. In the fraud detection domain, we have additional challenges when compared to the image domain. For example in image data, the normal samples are assumed to have shared features. However, in the fraud detection domain, even normal sessions can also exhibit large diversity. Therefore, learning separable representations for the open-set fraud detection task is challenging.

To address these challenges, we make the following contributions.

- We propose a novel framework called ConRo which is specifically designed for the open-set fraud detection task. Our ConRo framework operates in a scenario where only a few

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

malicious and a large amount of normal sessions are available.
- We propose a Long-Short Term Memory (LSTM) based session encoder which is trained by employing both supervised contrastive and DeepSVDD losses.
- We propose a data augmentation strategy to generate diverse potential malicious sessions in the representation space. We propose a strategy to filter generated false positive sessions.
- We present an empirical study on three benchmark fraud detection datasets: CERT [4], UMD-Wikipedia [8], and Openstack [2] in which, we show superior performance of our ConRo framework over state-of-the-art baselines.

2 CONRO FRAMEWORK

The user activities are modeled through activity sessions. Each session can consist of T user activities. Let $e_{i_t} (1 \leq t \leq T)$ denote the t^{th} activity of the i^{th} session. Each activity in a session is represented by an embedding vector, which can be trained based on the word-to-vector model. Let $\mathbf{x}_{i_t} \in \mathbb{R}^d$ denote the word-to-vector representation of activity e_{i_t}, where d denotes the number of representation dimensions. Here, $\mathbf{x}_i = \{\mathbf{x}_{i_t}\}_{t=1}^{T}$ denotes the *raw representation* of the i^{th} session. Let $\mathcal{X}$ and $\mathcal{Y}$ denote the raw input representation of sessions and label set, respectively. Here, $\mathcal{Y} = \{0, 1\}$ where $y = 0$ and $y = 1$ denote normal and malicious sessions, respectively. Let $\mathcal{D}$ denote the test set distribution over $\mathcal{X} \times \mathcal{Y}$ wherein, the test samples are drawn from $\mathcal{D}$. The training set $\mathcal{T}$ contains a large amount of normal and a few malicious sessions. Let $\mathcal{T}^0$ and $\mathcal{T}^1$ denote sets of normal and malicious sessions in $\mathcal{T}$, respectively. The malicious sessions sampled from $\mathcal{D}$ will also contain those unseen malicious sessions which are not present in $\mathcal{T}^1$.

Our ConRo framework has an encoder network that maps a session from its raw representation $\mathbf{x}$ to an *encoded representation* vector $\mathbf{z}$. We adopt LSTM as the foundation of our encoder to derive the encoded session representations. Our encoder consists of two hidden layers with the same dimensions. The hidden representations derived from the top layer of LSTM for the activities in the session $\mathbf{x}_i$ are denoted as $\{\mathbf{h}_{i_t}\}_{t=1}^{T}$. Here, $\mathbf{h}_{i_t} \in \mathbb{R}^d$. Then, the encoded session representation $\mathbf{z}_i \in \mathbb{R}^d$ is computed as $\mathbf{z}_i = \frac{1}{T}\sum_{t=1}^{T} \mathbf{h}_{i_t}$.

The main challenge which we are addressing is to design a procedure to obtain malicious sessions which are sampled from the test set distribution $\mathcal{D}$. To address this challenge, we construct potential malicious sessions which can be similar to malicious sessions sampled from $\mathcal{D}$. There are two main objectives for generating these potential malicious sessions: **MO1.** Malicious sessions usually form multiple clusters in the encoded representation space [5]. Malicious sessions belonging to the same cluster usually share close similarities. Hence, we need to generate potential malicious sessions which are similar to a seen malicious session $\mathbf{x}_i \in \mathcal{T}^1$. **MO2.** Suppose there are K malicious session clusters. However, the training set $\mathcal{T}$ might only contain $N (N < K)$ session clusters, and sessions belonging to remaining $K - N$ clusters are not present in $\mathcal{T}$. Note that the malicious sessions from these $K - N$ unseen clusters can diverge significantly from seen malicious sessions. We need to generate potential malicious sessions which belong to those $K - N$ clusters to effectively train our encoder. ConRo achieves these main objectives by employing a two stage encoder training procedure. We provide detailed descriptions of both these stages below.

2.1 First Stage

In the first stage, our encoder achieves two goals: (1) It learns shared features for normal sessions and learns to contrast normal sessions with seen malicious sessions in the encoded representation space. As a consequence, our encoder learns separable representations w.r.t to normal and seen malicious sessions. (2) It compresses normal session representations inside a minimum volume hypersphere in the encoded representation space. To achieve the first goal, we leverage the idea of supervised contrastive learning, which can learn separable representations w.r.t to normal and seen malicious sessions. Then, to achieve the second goal, we leverage the DeepSVDD loss [12], which pushes the normal samples inside a minimum volume hyper-sphere.

Supervised contrastive loss. We construct a training batch denoted as $S = \{\mathbf{x}_i\}_{i=1}^{R}$ by obtaining R random samples from $\mathcal{T}$. Since ConRo is specifically designed to operate on imbalanced training data, in order to effectively contrast malicious sessions with normal sessions, for each training batch S, we create a corresponding auxiliary batch $S^1 = \{\mathbf{x}_i^1\}_{i=1}^{M}$, by randomly sampling M malicious sessions from $\mathcal{T}^1$. We leverage a supervised contrastive loss function similar to the one presented by Khosla et al. [6]. This loss function is given by:

$$\mathcal{L}^{Sup} = \frac{1}{R}\sum_{i=1}^{R}(1 - y_i)\left(\frac{1}{|B^0(\mathbf{x}_i)|}\sum_{\mathbf{x}_p \in B^0(\mathbf{x}_i)} l\left(\mathbf{z}_i, \mathbf{z}_p, A(\mathbf{x}_i)\right)\right) \quad (1)$$

Here, the set $A(\mathbf{x}_i)$ is defined as $\left(S \cup S^1\right) - \{\mathbf{x}_i\}$, and the set $B^0(\mathbf{x}_i) = \left\{\mathbf{x}_p \in A(\mathbf{x}_i) | y_p = 0\right\}$ indicates samples $\mathbf{x}_p$ in $A(\mathbf{x}_i)$ with labels $y_p = 0$. The individual loss $l\left(\mathbf{z}_i, \mathbf{z}_p, A(\mathbf{x}_i)\right)$ between the pair $(\mathbf{x}_i, \mathbf{x}_p)$ is defined as:

$$l\left(\mathbf{z}_i, \mathbf{z}_p, A(\mathbf{x}_i)\right) = -\log\left(\frac{exp(\cos\left(\mathbf{z}_i \cdot \mathbf{z}_p\right)/\alpha)}{\sum_{\mathbf{x}_j \in A(\mathbf{x}_i)} exp(\cos\left(\mathbf{z}_i \cdot \mathbf{z}_j\right)/\alpha)}\right), \quad (2)$$

where α denotes the temperature parameter.

DeepSVDD loss. We leverage a DeepSVDD loss function which is similar to the one presented by Ruff et al. [12]. Let $\mathbf{v}_0 = \frac{1}{R_0}\sum_{i=1}^{R}(1 - y_i)\mathbf{z}_i$ denote the estimated center of normal sessions in the encoded representation space and $R_0 = \sum_{i=1}^{R}\mathbb{I}(y_i = 0)$, where $\mathbb{I}(\cdot)$ is an indicator function. This loss function is given by:

$$\mathcal{L}^{SV} = \frac{1}{R}\sum_{i=1}^{R}(1 - y_i)\left(||\mathbf{z}_i - \mathbf{v}_0||_2\right) \quad (3)$$

The loss function for the first stage is given by:

$$\mathcal{L}_1 = \mathcal{L}^{Sup} + \mathcal{L}^{SV} \quad (4)$$

To effectively address the session diversity challenge, we employ an alternating approach to optimize our encoder through $\mathcal{L}_1$. For each training batch $S = \{\mathbf{x}_i\}_{i=1}^{R}$, we first train our encoder through $\mathcal{L}^{Sup}$. As a result, we force the encoder to learn shared features for

normal sessions and contrast with seen malicious sessions in the encoded representation space (goal 1). Then, by using the same batch S, we train the encoder through $\mathcal{L}^{SV}$, which forces the encoder to compress normal session representations inside a minimum volume hyper-sphere in the encoded representation space (goal 2). Let r denote the radius of the normal session hyper-sphere obtained after first stage training.

2.2 Second Stage

First objective. We generate *similar potential malicious sessions* which are similar to a seen malicious session $\mathbf{x}_i \in \mathcal{T}^1$. Recently, Verma et al. [14] proposed a mix-up based data augmentation strategy for sequential data. Their augmentation strategy is inspired by the concept of *convex sets* and generates augmented samples that are similar to their original version. Specifically, they generate augmented samples by performing a mix-up operation on the encoded representations of original samples. Hence, we leverage a mix-up based augmentation strategy which is similar to the one presented by Verma et al. [14] for generating similar potential malicious sessions which are similar to a seen malicious session $\mathbf{x}_i \in \mathcal{T}^1$.

Let $\widehat{G}^1(\mathbf{x}_i)$ denote this set of generated similar potential malicious sessions. The set $\widehat{G}^1(\mathbf{x}_i)$ is defined as $\widehat{G}^1(\mathbf{x}_i) = \{\widehat{\mathbf{z}}|\widehat{\mathbf{z}} = \lambda_1 \mathbf{z}_i + (1-\lambda_1)\mathbf{z}_j\}$. Here, $\widehat{\mathbf{z}}$ denotes the encoded representation of a generated similar potential malicious session, $\mathbf{x}_j \in B^1(\mathbf{x}_i) = \left\{\mathbf{x}_p \in A(\mathbf{x}_i) | y_p = 1\right\}$ indicates samples $\mathbf{x}_p$ in $A(\mathbf{x}_i)$ with labels $y_p = 1$, λ_1 is sampled from the Uniform distribution $U(\beta_1, 1)$ where $\beta_1 \in [0, 1]$, and β_1 is set closer to 1 to ensure that generated potential malicious sessions have close similarities with $\mathbf{x}_i$.

Second objective. We generate *diverse potential malicious sessions* which can diverge significantly from a seen malicious session $\mathbf{x}_i \in \mathcal{T}^1$. Let $\widetilde{G}^1(\mathbf{x}_i)$ denote this set of generated potential malicious sessions. Our session augmentation strategy is inspired by the concept of *affine sets*. The set $\widetilde{G}^1(\mathbf{x}_i)$ is defined as $\widetilde{G}^1(\mathbf{x}_i) = \{\widetilde{\mathbf{z}}|\widetilde{\mathbf{z}} = \lambda_2 \mathbf{z}_i + (1-\lambda_2)\mathbf{z}_j, fp(\widetilde{\mathbf{z}}) = 0\}$. Here, $\widetilde{\mathbf{z}}$ denotes the encoded representation of a generated diverse potential malicious session, $\lambda_2 \sim U(-\beta_2, \beta_2)$, and $\beta_2 \in \mathbb{R}$. We treat β_2 as a hyper-parameter in our empirical studies. We filter a false positive through the function $fp(\cdot)$ as:

$$fp(\widetilde{\mathbf{z}}) = \begin{cases} 1, & \text{if } ||\widetilde{\mathbf{z}} - \mathbf{v}_0||_2 \leq r \\ 0, & otherwise \end{cases} \tag{5}$$

Second stage loss. We again leverage supervised contrastive loss to design our second stage loss function which is given by:

$$\mathcal{L}_2 = \frac{1}{R}\sum_{i=1}^{R}\left[y_i\left(\frac{1}{|D(\mathbf{x}_i)|}\sum_{\mathbf{x}_p \in D(\mathbf{x}_i)} l\left(\mathbf{z}_i, \mathbf{z}_p, C(\mathbf{x}_i)\right)\right)\right] \tag{6}$$

Here, $C(\mathbf{x}_i) = A(\mathbf{x}_i) \cup \widehat{G}^1(\mathbf{x}_i) \cup \widetilde{G}^1(\mathbf{x}_i)$, $D(\mathbf{x}_i) = B^1(\mathbf{x}_i) \cup \widehat{G}^1(\mathbf{x}_i) \cup \widetilde{G}^1(\mathbf{x}_i)$ and $l\left(\mathbf{z}_i, \mathbf{z}_p, C(\mathbf{x}_i)\right)$ denotes the individual loss between the pair $(\mathbf{x}_i, \mathbf{x}_p)$ corresponding to the malicious sessions defined as:

$$l\left(\mathbf{z}_i, \mathbf{z}_p, C(\mathbf{x}_i)\right) = -\log\left(\frac{exp(\cos\left(\mathbf{z}_i \cdot \mathbf{z}_p\right)/\alpha)}{\sum_{\mathbf{x}_j \in C(\mathbf{x}_i)} exp(\cos\left(\mathbf{z}_i \cdot \mathbf{z}_j\right)/\alpha)}\right) \tag{7}$$

Inference. After the second stage training, our encoder has learnt to push seen malicious sessions, similar and diverse potential malicious sessions closer in the encoded representation space, and as a consequence, all these sessions form a tight cluster in the encoded representation space. Normal sessions are also tightly clustered in the encoded representation space due to the effect of first stage training. Hence, we design our inference strategy by analyzing the proximities of a test case session to the centers of normal and malicious sessions in the encoded representation space. Let $\mathbf{v}_1$ denote the estimated center of malicious sessions in the encoded representation space, which is given by $\mathbf{v}_1 = \frac{1}{M}\sum_{i=1}^{M}\mathbf{z}_i^1$, where $\{\mathbf{x}_i^1\}_{i=1}^{M}$ denotes M randomly sampled malicious sessions from $\mathcal{T}^1$. For any test case session $\mathbf{x}$, ConRo predicts its label as:

$$label(\mathbf{x}) = \begin{cases} 1 & \text{if } ||\mathbf{z} - \mathbf{v}_1||_2 < ||\mathbf{z} - \mathbf{v}_0||_2 \\ 0 & otherwise \end{cases}$$

3 EXPERIMENTS

3.1 Experimental Setup

3.1.1 Datasets. We use three benchmark fraud detection datasets for our empirical study: CERT [4], UMD-Wikipedia [8], and OpenStack [2].

CERT [4]. The CERT dataset is a comprehensive dataset for insider threat detection. There are 48 malicious and 1,581,358 normal sessions. The insider sessions are chronologically recorded over 516 days. To avoid extreme training latency, we randomly sample 10,000 normal sessions from the first 460 days as the normal sessions in training set $\mathcal{T}$. Similarly, we randomly sample 500 normal sessions from 461 to 516 days to construct our test set. There are 5 types of malicious sessions. (1) *Logon*: The insider logs on a computer during weekends or on a weekday after work hours. (2) *Email*: The insider sends/views unexpected emails to/from external sources. (3) *HTTP*: The insider uploads/downloads organizational information to/from external malicious websites. (4) *Device*: The insider connects a device such as removable drives during weekends or on a weekday after work hours. (5) *File*: The insider manipulates organizational files with malicious intentions. We construct a biased training set corresponding to malicious sessions wherein, we include device, email, and file malicious session types in the training set and remaining two types in the test set. Specifically, we include 30 and 18 malicious sessions in the training and test sets, respectively.

UMD-Wikipedia [8]. This dataset consists of activity sessions of a set of users who have edited the Wikipedia website. In this dataset, there are 5486 normal and 4627 malicious sessions. We randomly sample 1000 normal sessions to construct the test set and include all the remaining 4486 normal sessions in the training set. For the malicious sessions, in-order to simulate open-set and imbalanced dataset scenario, we construct the training set by leveraging and suitably adapting the procedure utilized by Du et al. [3], which is described below. We calculate the appropriate number of malicious session clusters (K) in the available malicious sessions by using *silhouette coefficient analysis* [11]. From our empirical study, we get $K = 3$. Then, we randomly sample 70 and 10 malicious sessions from the first and second malicious session clusters, respectively, and include them in the training set. From the remaining malicious sessions, we randomly sample similar number of malicious sessions

from each of the 3 clusters to construct the test set which contains 500 malicious sessions.

OpenStack [2]. This dataset records the activity sessions of users who have used the OpenStack cloud services. In this dataset, there are 244,908 normal and 18,434 malicious sessions. We randomly sample 10,000 and 1000 normal sessions and include them in our training and test sets, respectively. For the malicious sessions, through silhouette coefficient analysis we get $k = 12$ malicious session clusters. We randomly sample 50 and 10 malicious sessions from the first and second malicious session clusters, respectively, and include them in the training set. From the remaining malicious sessions, we construct a test set having 120 malicious sessions by randomly sampling equal number of malicious sessions from each of the 12 malicious session clusters.

3.1.2 Training Details. By considering a user activity session as a sentence, we train the word-to-vector model to derive the activity representation. The minimum activity frequency is set as 1 since every activity is given importance in the design. To effectively train our session encoder, we set the number of dimensions of the activity and session representations as $d = 50$. Since we generate encoded session representation by averaging the output sequence of the LSTM model, we set the hidden layer size of LSTM to 50. The temperature parameter α shown in Equations 2 and 7 is set to its default value 1. We opt for medium sized training batches in order to avoid extreme memory requirements during encoder training. Specifically, we use 100 sessions (R) in each training batch. We set the size of the malicious session auxiliary batch (M) as 20. The sizes of potential malicious session batches $|\widehat{G}^1(\mathbf{x}_i)|$ and $|\widetilde{G}^1(\mathbf{x}_i)|$ are set as 20 and 200, respectively. For β_1, we set its value as 0.92 because it is supposed to be closer to 1. For β_2, we set its value as 4 in order to generate potential malicious sessions which are sufficiently diverse. We use the Adam optimizer [7] with a learning rate of 0.005 and we use 10 training epochs for both stages. We utilize three metrics to measure the anomaly detection performance: F_1, False Positive Rate (FPR), and Area Under the Receiver Operating Characteristics Curve (AUC-ROC). We report the mean and standard deviation of performance scores after 5 times of running.

3.1.3 Baselines. We compare our ConRo framework with four state-of-the-art baselines which were specifically designed for anomaly detection: DeepSVDD [12], DeepSAD [13], DevNet [10, 9], and Swan [1]. All these baselines operate in the setting where only a few anomalous samples are available. DeepSVDD, DeepSAD, and DevNet have been designed for closed set anomaly detection whereas, Swan has been designed for open-set anomaly detection. These baselines originally operate on image datasets and employ neural networks for image data such as CNN [13], ResNet-18 [1] etc. Hence, they cannot be directly applied for our fraud detection task which operates on sequential data. We replace their neural networks with our LSTM-based session encoder and adapt these baselines to our fraud detection task. We employ the same training set used for our ConRo to train all these baselines. We use 150 epochs to train all baselines. For Swan, the original augmentation technique is image-specific. Hence, we replace it with the augmentation technique proposed by Verma et al. [14]. Additionally, the unseen malicious sessions are detected in a residual representation space which is defined as: $\mathbf{v}_0 - \mathbf{z}$.

3.2 Experimental Results

3.2.1 Overall Comparison. The performance of our ConRo framework and baselines for all datasets are shown in Table 1. Clearly, our ConRo outperforms all baselines[1] w.r.t most of the performance metrics. These baselines do not learn effective class-specific shared features in the encoded representation space. Thus, due to the combined challenges of session diversity, dataset imbalance, and biased malicious training samples, they fail to provide noticeable results. However, ConRo addresses all these challenges effectively. It addresses the session diversity challenge through supervised contrastive learning. It addresses the dataset imbalance challenge by generating a large number of augmented/potential malicious sessions. Finally, it addresses the challenge of biased malicious training samples by generating diverse potential malicious sessions.

For the UMD-Wikipedia dataset, Swan noticeably outperforms our ConRo w.r.t FPR score. The mechanisms of ConRo and Swan are different. Specifically, Swan learns to identify unseen malicious sessions in a residual representation space ($\mathbf{v}_0 - \mathbf{z}$) whereas, ConRo learns to identify unseen malicious sessions by generating a large amount of diverse potential malicious sessions. In UMD-Wikipedia dataset, many normal sessions share close similarities with unseen malicious sessions. Therefore, ConRo identifies some of the test normal sessions sharing close similarities with test malicious sessions as malicious (false positive) which negatively impacts the FPR scores of ConRo. However, Swan does not specifically address the session diversity challenge in malicious sessions due to which, Swan under-performs against ConRo w.r.t F1 and AUCROC scores.

3.2.2 Ablation Analysis. We conduct the ablation analysis study on our ConRo framework by ablating the following main components: stage 1, $\mathcal{L}^{Sup}$, $\mathcal{L}^{SV}$, Alternating Optimization (AO), stage 2, $fp(\cdot)$, $\widehat{G}^1(\cdot)$, and $\widetilde{G}^1(\cdot)$. The ablation analysis results are shown in Table 2.

W/o stage 1. Mean F1 scores drop to 18.33 (CERT), 40.23 (UMD-Wikipedia), and 14.92 (Open-Stack). Stage 1 ensures that the encoder learns shared features for normal sessions. Without learning these shared features, the encoder fails to achieve tight class-specific clusters in the encoded representation space.

W/o $\mathcal{L}^{Sup}$. Mean F1 scores drop to 5.28 (CERT), 53.16 (UMD-Wikipedia), and 15.12 (Open-Stack). Both normal and malicious sessions typically exhibit large diversity and $\mathcal{L}^{Sup}$ is essential to address this session diversity challenge. We can see that there is a significant drop in F1 scores on CERT and Open-Stack datasets but not in the case for UMD-Wikiepdia dataset. We can attribute the reason to the different characteristics of these datasets. Addressing the session diversity challenge for the normal sessions is much more critical in both CERT and Open-Stack datasets than in the UMD-Wikipedia dataset.

W/o $\mathcal{L}^{SV}$. Mean F1 scores drop to 20.10 (CERT), 64.95 (UMD-Wikipedia), and 38.76 (Open-Stack). The DeepSVDD loss $\mathcal{L}^{SV}$ enables the encoder to push normal sessions in a minimum volume

[1]Since DevNet classifies all test sessions as normal, we have not shown its performance scores. Devnet does not employ any augmented malicious sessions for its training, so it cannot effectively address the dataset imbalance challenge.

Table 1: Performances of our ConRo and baselines (mean±std). The higher the better for F1 and AUC-ROC. The lower the better for FPR. The best values are bold highlighted.

Models	CERT			UMD-Wikipedia			Open-Stack		
	F1	FPR	AUC-ROC	F1	FPR	AUC-ROC	F1	FPR	AUC-ROC
DeepSVDD	14.67±4.1	14.30±1.8	62.29±4.8	33.23±1.7	44.90 ±1.4	46.47±0.8	32.27±0.7	42.10±1.4	79.02±0.7
DeepSAD	24.71±7.5	20.53±7.1	84.17±3.6	56.88±2.9	13.30±0.2	68.35±1.7	67.43±3.1	9.70±1.3	94.12±0.1
Swan	59.31±2.2	**0.0±0.0**	72.12±0.1	57.02±0.9	**0.0±0.0**	69.89±0.5	62.93±4.2	**0.0±0.0**	73.10±2.3
ConRo	**68.33±3.9**	2.20±0.5	**90.50±0.3**	**71.40±2.3**	31.50±2.1	**79.50±2.1**	**77.56±2.3**	5.80±0.8	**97.10±0.4**

Table 2: Ablation analysis results (mean±std).

Models	CERT			UMD-Wikipedia Dataset			Open-Stack		
	F1	FPR	AUC-ROC	F1	FPR	AUC-ROC	F1	FPR	AUC-ROC
w/o stage 1	18.33±1.2	25.10±0.3	78.56±1.1	40.23±1.3	28.37±1.9	55.55±1.1	14.92±1.7	47.14±2.3	49.43±2.9
w/o $\mathcal{L}^{Sup}$	5.28±0.2	48.10±1.8	45.44±0.9	53.16±2.3	25.10±1.9	64.65±1.9	15.12±1.4	46.90±2.7	49.78±2.4
w/o $\mathcal{L}^{SV}$	20.10±1.1	27.05±1.3	83.72±0.7	64.95±0.8	18.85±6.1	73.72±0.5	38.76±0.8	31.60±1.1	84.20±0.4
w/o AO	8.99±0.4	68.46±3.1	62.98±1.6	52.04±1.4	80.30±1.4	55.68±1.9	14.08±2.1	26.16±1.4	50.59±2.3
w/o stage 2	42.86±1.1	0.0±0.0	63.84±0.1	60.90±1.1	23.85±1.6	70.42±0.6	46.11±3.4	0.0±0.0	65.40±1.6
w/o $fp(\cdot)$	31.32±5.1	26.66±6.3	72.77±3.1	59.38±1.8	65.52±4.2	65.95±2.6	37.50±3.7	49.60±3.6	48.16±3.5
w/o $\widehat{\mathbf{G}}^{\mathbf{1}}(\cdot)$	55.92±1.2	4.13±0.3	89.49±0.2	65.58±2.6	31.20±0.3	74.04±2.3	67.57±1.1	9.60±0.4	95.20±0.2
w/o $\widetilde{\mathbf{G}}^{\mathbf{1}}(\cdot)$	44.17±1.9	7.10±0.6	88.16±0.3	63.40±0.8	31.70±4.7	72.10±0.6	52.26±1.6	18.10±1.4	90.61±0.2

hyper-sphere in the encoded representation space. Without this topological effect, the efficacy of stage 2 reduces because the generated diverse potential malicious sessions do not effectively cover unseen malicious sessions.

W/o AO . By employing the joint optimization approach, mean F1 scores drop to 8.99 (CERT), 52.04 (UMD-Wikipedia), and 14.08 (Open-Stack). Optimizing DeepSVDD objective ($\mathcal{L}^{SV}$) can yield maximum benefits only when the input normal sessions have considerable shared features in the encoded representation space. Here, we jointly optimize both supervised contrastive ($\mathcal{L}^{Sup}$) and DeepSVDD objectives, and we do not specifically provide normal sessions having considerable shared features in the encoded representation space as inputs to the DeepSVDD objective. As a consequence, we can observe a significant drop in the performance.

W/o stage 2 . Mean F1 scores drop to 42.86 (CERT), 60.90 (UMD-Wikipedia), and 46.11 (Open-Stack). In stage 1 training, our encoder learns to contrast normal sessions with few available malicious sessions having limited diversity. Stage 2 generates diverse potential malicious sessions which can be similar to unseen malicious sessions w.r.t their encoded representations. As a consequence, our encoder can learn effective separable encoded representations w.r.t open-set fraud detection task.

W/o fp$(\cdot)$. Mean F1 scores drop to 31.32 (CERT), 59.38 (UMD-Wikipedia), and 37.50 (Open-Stack). Without employing $fp(\cdot)$, the encoder learns to push malicious sessions and those potential malicious sessions which are false positives, closer in the encoded representation space. Due to this improper learning effect, the encoder does not achieve effective separable encoded representations.

W/o $\widehat{\mathbf{G}}^{\mathbf{1}}(\cdot)$. Without employing the similar potential malicious sessions during the second stage encoder training, mean F1 scores drop to 55.92 (CERT), 65.58 (UMD-Wikipedia), and 67.57 (Open-Stack). Generating similar potential malicious sessions which are similar to a seen malicious session aids the encoder to learn more effective separable representations. Additionally, $\widehat{\mathbf{G}}^{\mathbf{1}}(\cdot)$ aids in addressing the dataset imbalance challenge.

W/o $\widetilde{\mathbf{G}}^{\mathbf{1}}(\cdot)$. Without employing the diverse potential malicious sessions during the second stage encoder training, mean F1 scores drop to 44.17 (CERT), 63.40 (UMD-Wikipedia), and 52.26 (Open-Stack). Generating diverse potential malicious sessions which can be similar to unseen malicious sessions in the encoded representation space, aids the encoder to effectively contrast normal sessions with unseen malicious sessions, and can learn separable representations w.r.t open-set fraud detection task.

4 CONCLUSION

In this work, we have developed a robust and open-set fraud detection framework called ConRo, which is specifically designed to operate in the scenario where only a few malicious sessions having limited diversity are available for training. We developed a training procedure for ConRo to learn separable session representations by employing effective data augmentation strategies and by the combined effect of supervised contrastive and DeepSVDD losses. The empirical study on three benchmark datasets demonstrated that our ConRo can outperform state-of-the-art baselines. In our future work, we plan to extend ConRo to address specific distribution shift scenarios such as *sample selection bias*. We will study how to integrate bias correction approaches with supervised contrastive learning.

ACKNOWLEDGEMENT

This work was supported in part by NSF grants 1920920, 1946391 and 2103829.

REFERENCES

[1] Choubo Ding, Guansong Pang, and Chunhua Shen. 2022. Catching both gray and black swans: open-set supervised anomaly detection. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[2] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. 2017. Deeplog: anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, CCS*.

[3] Wei Du and Xintao Wu. 2021. Fair and robust classification under sample selection bias. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*.

[4] Joshua Glasser and Brian Lindauer. 2013. Bridging the gap: A pragmatic approach to generating insider threat data. In *IEEE Symposium on Security and Privacy Workshops*.

[5] Hyunjun Ju, Dongha Lee, Junyoung Hwang, Junghyun Namkung, and Hwanjo Yu. 2020. Pumad: pu metric learning for anomaly detection. *Information Sciences*.

[6] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems*.

[7] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*.

[8] Srijan Kumar, Francesca Spezzano, and V.S. Subrahmanian. 2015. Vews: a wikipedia vandal early warning system. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[9] Guansong Pang, Choubo Ding, Chunhua Shen, and Anton van den Hengel. 2021. Explainable deep few-shot anomaly detection with deviation networks. *CoRR*, abs/2108.00462.

[10] Guansong Pang, Chunhua Shen, and Anton van den Hengel. 2019. Deep anomaly detection with deviation networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

[11] Peter J. Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*.

[12] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. 2018. Deep one-class classification. In *Proceedings of the 35th International Conference on Machine Learning*.

[13] Lukas Ruff, Robert A. Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. 2020. Deep semi-supervised anomaly detection. In *8th International Conference on Learning Representations*.

[14] Vikas Verma, Thang Luong, Kenji Kawaguchi, Hieu Pham, and Quoc V. Le. 2021. Towards domain-agnostic contrastive learning. In *Proceedings of the 38th International Conference on Machine Learning*.

[15] Shuhan Yuan and Xintao Wu. 2021. Deep learning for insider threat detection: review, challenges and opportunities. *Computers & Security*.

[16] Shuhan Yuan, Panpan Zheng, Xintao Wu, and Hanghang Tong. 2020. Few-shot insider threat detection. In *The 29th ACM International Conference on Information and Knowledge Management*.