



Few-shot anomaly detection and classification through reinforced data selection with a combinatorial reward

Xiao Han¹ · Depeng Xu² · Shuhan Yuan¹ · Xintao Wu³

Received: 23 January 2023 / Revised: 24 June 2025 / Accepted: 4 August 2025 /

Published online: 30 August 2025

© The Author(s) 2025

Abstract

Due to the scarcity of anomalies, deep anomaly detection models are typically trained in an unsupervised or semi-supervised manner, depending on the availability of a small number of labeled samples. Currently, most unsupervised approaches detect anomalies by identifying the deviant patterns from normal samples, and some semi-supervised studies also use labeled anomalies to improve performance. However, few studies have focused on how to take advantage of potential anomalies in an easily obtained and large-scale unlabeled dataset. Meanwhile, in a semi-supervised setting, although we assume there will be a small number of labeled anomalies, the task of anomaly classification is under-exploited, which is important for domain experts. In this work, we focus on the problem of anomaly detection and classification with limited labeled samples and a large number of unlabeled samples. To this end, we develop a few-shot anomaly detection and classification model based on reinforced data selection with a combinatorial reward, called FADScr. FADScr iteratively improves performance by exploring the unlabeled dataset and selects informative samples to augment the training set to enhance both anomaly detection and classification. Experimental results show that our proposed framework is able to improve the performance of anomaly detection and classification with only a few labeled samples initially.

Keywords Anomaly detection · Few-shot learning · Reinforcement learning

✉ Shuhan Yuan
shuhan.yuan@usu.edu

Xiao Han
xiao.han@usu.edu

Depeng Xu
depeng.xu@uncc.edu

Xintao Wu
xintaowu@uark.edu

¹ School of Computing, Utah State University, Logan, UT, USA

² Software and Information Systems Department, University of North Carolina at Charlotte, Charlotte, NC, USA

³ Electrical Engineering and Computer Science Department, University of Arkansas, Fayetteville, AR, USA

1 Introduction

Anomaly detection indicates the detection of data samples that significantly deviate from the majority of data [7, 41]. Because of the extensive demand in a wide spectrum of applications, such as external and internal threats in cyberspace, anomaly detection has become an increasingly important research task.

1.1 Motivation and challenges

Due to the small number of anomalies, classical supervised learning algorithms cannot be employed. Currently, the majority of approaches are trained in an unsupervised or semi-supervised learning manner [6–8, 41, 63]. However, one limitation of existing anomaly detection approaches is that existing approaches cannot further classify anomalies into specific anomaly categories. In many real-world scenarios, understanding the types of anomalies is critical. For example, for the malicious insider threat detection task, the malicious insider conducts a variety of malicious activities, such as using a thumb drive to steal data before leaving the company, logging into another user's machine and searching for interesting files, or uploading documents to Dropbox for personal gain. Anomaly detection aims to label all malicious activities as anomalies, which is good. Meanwhile, anomaly classification further predicts the malicious activities into corresponding categories, which can provide more insight to the domain experts so that defense strategies can be designed precisely.

However, due to the scarcity of anomalies, the number of anomalies in each class is even smaller. Hence, it is extremely hard to train a multiclass classifier in a traditional supervised manner. Recently, to learn from a limited number of labeled samples, few-shot learning as a special type of machine learning has become an emerging research topic, which aims to learn classifiers given only a few labeled samples of each class [58]. One common strategy of few-shot learning is to project limited samples into a smaller embedding space so that similar samples are grouped together while dissimilar samples are separated [3, 54, 57, 58]. However, in the anomaly detection scenario, such cluster assumption only holds for normal samples since normal data are similar. For anomalies, by only having a few samples, it is hard to build a cluster to represent a class of anomalies, especially considering that anomalies are much more diverse. As a result, current few-shot anomaly detection approaches only work on distinguishing anomalies from normal samples and cannot further divide the anomalies into fine-grained classes [11, 37, 42, 62]. Hence, how to leverage the powerful few-shot learning models for anomaly detection and classification is still under-exploited.

1.2 Proposed approach: FADScr

In this work, by following real-world scenarios, we assume that we have plenty of labeled normal samples and limited abnormal samples in each anomaly class, as well as large-scale unlabeled samples. To achieve fine-grained anomaly detection based on few-shot learning, we propose a framework, called the few-shot anomaly detection and classification model through reinforced data selection with a combinatorial reward (FADScr), to leverage the large-scale unlabeled dataset to progressively improve the few-shot learner for anomaly detection and classification.

We assume that the unlabeled dataset follows the data distribution in the real world and consists of a large number of normal samples and a few anomalies. Initially, we train a few-shot learning model on a few labeled anomalies. Then, we iteratively update the few-shot

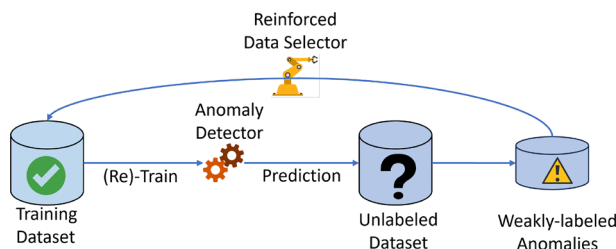


Fig. 1 FADScr iteratively improving the performance of anomaly detection and classification

model by selecting potential anomalies from unlabeled samples to augment the labeled training set. Each iteration consists of two steps: data selection and model retraining. In the first step, we apply the current few-shot model to predict anomalies on the unlabeled dataset. The prediction outcomes can be considered as weakly labeled samples, as their predicted labels can be either accurate or inaccurate. We then apply our proposed reinforcement learning-based data selection to identify the weakly labeled samples with high quality. The chosen samples have a high chance of being correctly labeled and, hence, are used to compose an augmentation set. After that, we build a new augmented training dataset by combining the existing labeled dataset with the newly generated augmentation set. In the second step, we retrain the few-shot learning model on the augmented training dataset to improve its performance. We expect the performance of FADScr to keep improving as the training procedure moves forward. Figure 1 shows the pipeline of FADScr.

The key step of FADScr is to select weakly labeled samples to augment the training set. If the misclassified samples are selected for the training set, the anomaly detection model can even be damaged. Hence, we develop a novel reinforcement learning approach to select high-quality samples. Specifically, we consider the high-quality samples from two perspectives. First, we expect the high-quality, weakly labeled samples to be able to improve the performance of anomaly detection. Second, the selected samples can further reduce the uncertainties of predicted output, meaning that the model should be confident in the predictions (high predicted probabilities). To this end, we design the reward function based on the performance improvement of the anomaly detection model and the uncertainty reduction on predicted samples.

1.3 Contributions

The contributions of this work can be summarized as follows. First, we propose a novel framework to progressively improve the few-shot model for anomaly detection and classification through reinforced data selection from an unlabeled dataset. The few-shot model can achieve fine-grained prediction given a small number of labeled anomalies in each class. Second, we propose a reinforcement learning-based data selection strategy that can select correctly labeled samples from a weakly labeled set to augment the training set and further improve the performance of the few-shot model. Third, experimental results show that FADScr can achieve state-of-the-art performance in anomaly detection and few-shot anomaly classification.

The work is significantly extended from our previous work [21]. The novel contribution with respect to our previous work is that we integrate an active learning-inspired strategy into few-shot anomaly detection by selecting high-uncertainty samples for augmentation.

Unlike previous work that may prioritize easy samples, this approach explicitly quantifies uncertainty using predicted class probabilities and prioritizes hard samples (i.e., those with low predicted probabilities for their true class). To ensure these hard samples contribute meaningfully to model improvement, a novel reward function is introduced, which evaluates probability changes on hard samples after training on the augmented set. The selection of challenging examples significantly updates class prototypes in prototypical networks, ultimately enhancing model robustness and performance in few-shot settings. The comparison of our previous approach and FADScr in anomaly detection and classification, shown in Table 4, demonstrates the advantage of FADScr.

1.4 Organization of the paper

The remainder of this paper is structured as follows. Section 2 reviews related work on anomaly detection. Section 3 presents our proposed approach, FADScr, for improving anomaly detection and classification through a reinforced data selection mechanism. Section 4 describes the experimental results of our approach and baselines on three datasets. Section 5 concludes the paper and outlines future research directions.

2 Related work

The tasks of anomaly detection have been studied for decades. In recent years, various learning-based anomaly detection approaches have been proposed [7, 8, 41]. As anomalies are scarce, unsupervised and semi-supervised approaches are widely used to detect anomalies because manually labeling a large number of abnormal samples is time-consuming and labor-intensive [5, 14, 23, 38, 40, 47, 65].

Unsupervised anomaly detection. The unsupervised anomaly detection approaches usually assume the availability of normal samples. The basic idea is to capture the normal patterns, and the anomalies can then be detected with highly deviant patterns. The reconstruction-based idea is the most commonly used idea for anomaly detection. The key idea is that after training to reconstruct normal samples, anomalies can be highlighted with high reconstruction errors. Traditionally, vanilla auto-encoder-based structures are widely adopted for reconstructing the input [2, 67]. Besides reconstruction, generative models, such as Generative Adversarial Networks (GAN) [18] or Variational Autoencoder (VAE) [29], are also adopted for anomaly detection. Similar to the reconstruction-based idea, generative models learn to generate normal samples while anomalies are then detected with a significant difference in the learned representations. For example, AnoGAN [49] and its variances, such as f-AnoGAN [50], leverage both reconstruction error from the generator and discriminator loss as the anomaly scores for anomaly detection. Recently, some advanced generative models, such as diffusion models [61], have also been adopted for anomaly detection, especially in the image anomaly detection domain [22, 59].

However, recent studies also demonstrate that only observing the normal samples for training could lead to a high false-positive rate due to a lack of prior knowledge of true anomalies [43, 64].

Few-shot anomaly detection. Few-shot learning models, which usually learn classifiers given only a few labeled samples of each class, have demonstrated great performance in computer vision and natural language processing [58]. Few-shot learning is able to learn models for rare cases and can reduce data-collecting efforts. Hence, applying the few-shot learning techniques

for anomaly detection is a natural fit. Recently, several few-shot learning approaches for anomaly detection have been proposed [11, 30, 33, 37, 68].

In many real-world scenarios, a small number of labeled abnormal samples, e.g., from domain experts, are available. Hence, semi-supervised anomaly detection approaches, which assume the availability of labeled samples as well as large-scale unlabeled samples, are commonly used to boost the performance of anomaly detection by leveraging the limited anomalies [1, 20, 23, 47, 56]. Therefore, one of the few-shot anomaly detection settings assumes a few anomalies available for detection. DeepSAD is a classical work that leverages labeled normal and abnormal samples for improving the anomaly detection performance [47]. Recent studies [11, 12, 44] further focus on leveraging a few labeled anomalies to identify similar abnormal samples.

Another few-shot anomaly detection setting assumes only a limited number of normal samples are available and aims to identify the corresponding anomalies [4, 20, 24, 26, 27, 45, 53]. Traditionally, these models adopt one model to identify anomalies in one class of images, and various data augmentation techniques are commonly adopted for improving the performance [24, 45, 53]. Recently, some approaches can identify anomalies from multiple categories via domain knowledge or teacher models [20, 34].

The existing studies can only detect anomalies but cannot achieve fine-grained classification. In this work, we aim to improve the few-shot learner for anomaly detection and classification progressively.

Reinforcement learning for anomaly detection. In literature, a few studies also leverage the reinforcement learning technique for anomaly detection [15, 25, 32, 40, 43, 64, 66]. [64] consider anomaly detection in a data stream setting as a sequential decision process. The agent aims to select true anomalies from data streams for human judgment within a budget limit. [40] propose an inverse reinforcement learning model for sequential anomaly detection in an unsupervised setting by learning an implicit reward function to guide the agent. [43] propose a reinforcement learning framework to identify unknown anomalies by training an anomaly detection agent based on a few classes of known anomalies.

In this work, we propose a reinforcement learning framework to augment the training dataset by exploring large-scale, unlabeled samples. The agent in our approach is trained to select anomalies that can improve the performance of the few-shot learning model for anomaly detection and classification.

3 Proposed approach

3.1 Problem definition

Let $\mathcal{L} = \{(x_i, y_i)\}_{i=1}^{N_L}$ be a labeled dataset, where x_i indicates the i -th sample, while $y_i \in \{0, 1, \dots, K\}$ indicates the corresponding label. Particularly, $y_i = 0$ indicates a normal sample while $y_i \in \{1, \dots, K\}$ denotes a class of anomalies. Considering that it is usually feasible to have a large number of normal samples in the anomaly detection scenario, the labeled dataset can be decomposed as $\mathcal{L} = \mathcal{L}_N \cup \mathcal{L}_A$ with $|\mathcal{L}_N| > |\mathcal{L}_A|$, where $\mathcal{L}_N$ only consists of normal samples and $\mathcal{L}_A$ includes few samples in each anomaly class.

With such a small number of abnormal samples, it is hard to train an accurate anomaly detection and classification model. To tackle this challenge, in this work, besides a labeled dataset $\mathcal{L}$, we further leverage an unlabeled dataset $\mathcal{U} = \{x_j\}_{j=1}^{N_U}$. The goal is to learn a few-shot anomaly detection and classification model that can leverage the knowledge of

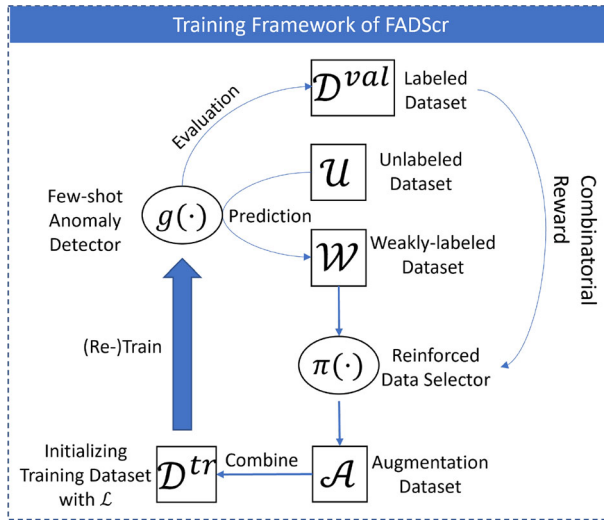


Fig. 2 Training framework of FADScr

the unlabeled dataset $\mathcal{U}$, especially the potential anomalies in $\mathcal{U}$, to maximally improve the performance of the model on anomaly detection and classification.

3.2 Framework overview

In this work, we propose a few-shot anomaly detection and classification model through reinforced data selection with a combinatorial reward (FADScr), which is able to gradually enhance the performance of the few-shot learning model by exploiting the unlabeled dataset $\mathcal{U}$.

In particular, FADScr first trains a few-shot learning model on an initial training dataset with only labeled samples $\mathcal{D} = \mathcal{L}$. Then, FADScr iteratively improves the model by selecting samples from the unlabeled dataset $\mathcal{U}$ to augment the training dataset. We employ the reinforcement learning technique for data selection. In each training iteration, we first use the current few-shot model to predict the label $\hat{y}_j$ for each sample x_j in $\mathcal{U}$, thus producing a weakly labeled dataset $\mathcal{W} = \{(x_j, \hat{y}_j)\}_{j=1}^{N_U}$. Then, we train a reinforcement learning agent to select the weakly labeled samples that have high chances to be accurate into an augmentation set $\mathcal{A} = \{(x_j, \hat{y}_j) | a_j = 1\}_{j=1}^{N_U}$, where $a_j \in \{0, 1\}$ indicates whether the sample $(x_j, \hat{y}_j)$ is selected by the agent or not. We combine the augmentation dataset $\mathcal{A}$ with the existing training dataset to compose the new training dataset, i.e., $\mathcal{D} = \mathcal{D} \cup \mathcal{A}$, and then retrain the few-shot model on the new training dataset. We expect that the performance of the few-shot model will be improved with those augmented samples. Therefore, as the training iteration moves forward, the few-shot anomaly detection model can be improved progressively. The overview of FADScr is shown in Fig. 2.

3.3 Prototypical network

In this work, we adopt the prototypical network [54] as our base few-shot learning model. Following the typical process for training the few-shot learning model, we first randomly select N_S and N_Q samples from the training dataset $\mathcal{D}$ to compose the support set $\mathcal{S}$ and query set $\mathcal{Q}$, respectively.

The prototypical network learns an embedding function $g(\cdot)$ to map each sample x_i to an embedding space, denoted as $\mathbf{x}_i = g(x_i)$. Based on the support set, the prototype representation for each class can be derived by a mean operation:

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(x_i, y_i) \in S_k} g(x_i), \quad (1)$$

where S_k indicates the subset of samples in the support set $\mathcal{S}$ with the class k . Then, given a distance function $d(\cdot)$, the prototypical network predicts the distribution of classes for a query point $x \in \mathcal{Q}$ based on a softmax over distances to the prototypes:

$$p(y = k|\mathbf{x}) = \frac{\exp(-d(g(\mathbf{x}), \mathbf{c}_k))}{\sum_{k'} \exp(-d(g(\mathbf{x}), \mathbf{c}_{k'}))}. \quad (2)$$

The objective function of the prototypical network is the negative log-probability $L = -\log p(y = k|\mathbf{x})$ of the true class k . The training objective is to make the samples from the same class have small distances to the class prototype $\mathbf{c}_k$. A test sample can be labeled based on the label of its nearest prototype in the embedding space.

3.4 Reinforced data selection

As the labeled dataset $\mathcal{L}$ only has a very small number of anomalies, the performance of the initial prototypical network trained on $\mathcal{L}$ still has room for improvement if more training samples can be incorporated. To this end, we propose to exploit the unlabeled dataset $\mathcal{U}$. We first use the current prototypical network to predict the labels of samples in $\mathcal{U}$ and get a weakly labeled dataset $\mathcal{W}$, which consists of both correctly and incorrectly labeled samples. If we simply adopt $\mathcal{W}$ to augment the training set, due to the noisy supervised signals, the performance of the prototypical network could be worse. Hence, we propose a reinforcement learning-based data selection method to select samples from $\mathcal{W}$ based on their reliability to compose an augmentation set $\mathcal{A}$. Once the reinforced data selection agent is capable of selecting the correctly labeled samples to augment the training set, the performance of the few-shot learner can be improved. Therefore, the reward to the data selection agent is designed based on the performance of the few-shot learner on an unobserved validation set. The key components of the reinforcement learning framework are described below.

State. For each state s_j with weakly labeled sample $(x_j, \hat{y}_j) \in \mathcal{W}$, its state representation $\mathbf{s}_j$ is defined as the concatenation of the embedding vector $\mathbf{x}_j = g(x_j)$ and the distance value d_j to the closest prototype, i.e., $\mathbf{s}_j = [\mathbf{x}_j, d_j]$.

Action. The data selection agent then needs to take an action whether to select this sample $(x_j, \hat{y}_j)$ into the augmentation set $\mathcal{A}$ based on the state representation $\mathbf{s}_j$. Specifically, $a_j = 0$ means the sample $(x_j, \hat{y}_j)$ will be rejected, while $a_j = 1$ indicates the sample will be selected and added to the augmentation set $\mathcal{A}$.

Policy network. The data selection agent makes decisions about whether to select a weakly labeled sample based on a policy network $\pi_\theta(\cdot)$. We adopt a neural network to parameterize

the policy network that takes the state representation as input and outputs the probability of the action, $p(a_j|s_j) = \pi_\theta(s_j)$. The action a_j is then sampled based on $p(a_j|s_j)$.

Rewards. The policy network is trained with guidance from the reward function. We define the reward based on the performance of the few-shot learning model on an unseen validation set. In our task, we aim to detect anomalies as well as classify abnormal samples into different classes. Therefore, we design the reward function based on the performance of anomaly detection and classification, i.e., an anomaly detection reward r^d and a classification reward r^c . Specifically, we adopt the F1 score as the metric to evaluate the performance of anomaly detection and the macro-F1 score over all classes to evaluate the performance of classification. In our scenario, if wrongly labeled samples are selected to compose the training dataset, we can expect that the performance of the few-shot learning model will be damaged. On the other hand, if the samples with the correct labels are selected, the performance of the model can be improved. Hence, the reward is designed based on the performance change after the model is trained on an augmentation set.

Moreover, inspired by active learning with the strategy of labeling uncertain samples, we expect the agent to select the samples with high uncertainties to the augmentation set so that the performance of the few-shot learning model for anomaly detection and classification can be further improved. Here, the uncertainty of a sample is quantified by its predicted probabilities $p(y = k|\mathbf{x})$. When the predicted probability of a sample is less than a threshold, i.e., $p(y = k|\mathbf{x}) < \tau$, we consider the sample as a hard sample. Otherwise, the samples are easy samples. In experiments, we set $\tau = 0.5$, less than this threshold indicating that the model is uncertain about the sample. If the agent only selects easy samples into the augmentation set, for prototypical networks, the center $\mathbf{c}_k$ of the class (defined in Eq. 1) cannot be updated significantly, meaning that the hard samples still have low predicted probabilities. Therefore, we also design a reward r^p based on probability changes on hard samples after the model is trained on an augmentation set.

Specifically, in each episode, we first calculate the F1, macro-F1 scores, and predicted probabilities $p(y = k|\mathbf{x})$ on validation samples by the current prototypical network. We denote the prototypical network at the beginning of each episode as $g_0(\cdot)$ and the corresponding F1, macro-F1 scores achieved by $g_0(\cdot)$ as $F1_0$ and $MacroF1_0$. Meanwhile, to evaluate whether the prototypical network can reduce the prediction uncertainties, we also select hard samples, i.e., $p(y = k|\mathbf{x}) < \tau$, from the validation set and use the mean probabilities as the uncertainty score, denoted as Pr_0 .

Then, we work on improving $g_0(\cdot)$ by composing an augmentation set. To this end, first, we randomly sample a set of batches $\mathcal{B} = \{B_l\}_{l=1}^L$ from $\mathcal{W}$, where each batch B_l consists of a number of samples. Given a batch B_l , for each sample in B_l , we sample an action a_j (select to the augmentation set or not) based on the policy $p(a|s_j) = \pi_\theta(s_j)$. Then, we can compose the augmentation set $\mathcal{A}_l$ from the batch B_l . After combining $\mathcal{A}_l$ with $\mathcal{D}$, we update the prototypical network, denoted as $g_l(\cdot)$, and further evaluate $g_l(\cdot)$ on a validation set to derive F1, macro-F1 scores, and probabilities, denoted as $F1_l$, $MacroF1_l$, Pr_l .

Finally, the reward for the agent in a batch can be calculated as the differences of F1 scores and probabilities, $r_l^d = F1_l - F1_0$, $r_l^c = MacroF1_l - MacroF1_0$, and $r_l^p = Pr_l - Pr_0$. The overall reward function in a batch is formulated as:

$$r_l = r_l^d + \alpha \cdot r_l^c + \beta \cdot r_l^p, \quad (3)$$

where α and β are hyperparameters to balance anomaly detection, classification tasks, and the reward for reducing uncertain samples.

Optimization. The data selection agent is trained based on the actor-critic algorithm [31]. The output of the actor is the probability of two actions (select or not), while the output of the critic is the predicted reward value based on the current state. Both actor and critic are parameterized by feedforward neural networks. The goal of the agent is to maximize the rewards, which is defined as:

$$\mathcal{J}(\theta) = \mathbb{E}_{\pi_{\theta}}[r_l]. \quad (4)$$

The parameter θ in policy network π_{θ} is trained based on policy gradient [55]:

$$\theta \leftarrow \theta + \eta \nabla_{\theta} \mathcal{J}(\theta), \quad (5)$$

where η indicates the learning rate. The gradient for a batch of weakly labeled samples can be approximated by [31]

$$\nabla_{\theta} \mathcal{J}(\theta) = \frac{1}{|B_l|} \sum_{j=1}^{|B_l|} \nabla_{\theta} \log \pi_{\theta}(\mathbf{s}_j) (r_l - V_{\varphi}(\mathbf{s}_j)), \quad (6)$$

where $|B_l|$ indicates the number of samples in a batch B_l ; $V_{\varphi}(\mathbf{s}_j)$ is the expected reward from a critic network $V_{\varphi}(\cdot)$ parameterized by φ . The structure of the critic network is similar to the policy network with the last layer as a regression function. The critic network is designed to estimate the expected reward and hence updated according to the cumulative difference between the real reward r_l and the predicted value $V_{\varphi}(\mathbf{s}_j)$,

$$\mathcal{L}(\varphi) = \frac{1}{|B_l|} \sum_{j=1}^{|B_l|} |r_l - V_{\varphi}(\mathbf{s}_j)|. \quad (7)$$

The parameters φ in $V_{\varphi}(\mathbf{s}_j)$ can be optimized by:

$$\varphi = \varphi - \eta' \nabla_{\varphi} \mathcal{L}(\varphi), \quad (8)$$

where η' indicates the learning rate for updating the critic network.

3.5 Training details

General training process. Algorithm 1 shows the pseudo-code of the training process. Given a small labeled dataset $\mathcal{L}$ and an unlabeled dataset $\mathcal{U}$, we first initialize an augmentation dataset $\mathcal{A}$ as an empty set (Line 1) and a training dataset $\mathcal{D}$ as the small labeled dataset $\mathcal{L}$ (Line 2). In each episode, we split $\mathcal{D}$ into two parts, $\mathcal{D}^{tr}$ as the real training set and $\mathcal{D}^{val}$ as the unobserved validation set (Line 4), combine $\mathcal{D}^{tr}$ with $\mathcal{A}$ as the new training set (Line 5), and use it to train the prototypical network $g(\cdot)$ (Line 6). We deploy the prototypical network to predict labels of samples in $\mathcal{U}$ and get a weakly labeled data set $\mathcal{W}$ (Line 7). We then apply a reinforced data selection algorithm to generate a batch of samples as the augmentation set $\mathcal{A}_{ep}$ from $\mathcal{W}$ (Line 8). After obtaining $\mathcal{A}$, we remove $\mathcal{A}_{ep}$ from the unlabeled dataset $\mathcal{U}$ (Line 9) and add $\mathcal{A}_{ep}$ to $\mathcal{A}$ to form a new augmentation set for the next iteration (Line 10).

With the increase in training episodes, the size of the augmentation dataset $\mathcal{A}$ keeps increasing, and the performance of the prototypical network becomes better. After finishing all the training iterations, we further combine labeled dataset $\mathcal{D}$ with the augmentation dataset $\mathcal{A}$, retrain the prototypical network from scratch based on the combined dataset (Line 11), and deploy it for anomaly detection and classification.

Reinforced data selection. The key step for improving the performance of the prototypical network is the reinforced data selection algorithm (Lines 13–29), which is to select the

Algorithm 1 The training process of FADScr

Input : Labeled set $\mathcal{L}$, unlabeled set $\mathcal{U}$
Output: Prototypical network $g(\cdot)$

- 1 Initialize an augmentation set $\mathcal{A} \leftarrow \emptyset$
- 2 Initialize a training dataset $\mathcal{D} \leftarrow \mathcal{L}$
- 3 **for** $t = 0$ **to** T **do**
- 4 Split $\mathcal{D}$ to $\mathcal{D}^{tr}$ and $\mathcal{D}^{val}$
- 5 $\mathcal{D}^{tr} \leftarrow \mathcal{D}^{tr} \cup \mathcal{A}$
- 6 Train prototypical network $g(\cdot)$ on $\mathcal{D}^{tr}$
- 7 Predict labels on $\mathcal{U}$ and get $\mathcal{W}$ based on $g(\cdot)$
- 8 $\mathcal{A}_{ep} \leftarrow \text{Data_Selection}(g(\cdot), \mathcal{W}, \mathcal{D}^{tr}, \mathcal{D}^{val})$
- 9 Remove selected samples $\mathcal{U} \leftarrow \mathcal{U} / \mathcal{A}_{ep}$
- 10 Increase the augmentation dataset $\mathcal{A} \leftarrow \mathcal{A} \cup \mathcal{A}_{ep}$
- 11 Train prototypical network $g(\cdot)$ on $\mathcal{D} \cup \mathcal{A}$
- 12 **return** prototypical network $g(\cdot)$

- 13 **Function** $\text{Data_Selection}(g_0(\cdot), \mathcal{W}, \mathcal{D}^{tr}, \mathcal{D}^{val})$:
- 14 Apply $g_0(\cdot)$ on $\mathcal{D}^{val}$ to derive $(F1_0, MacroF1_0, Pr_0)$
- 15 Generate subsets $\mathcal{B} = \{B_l\}_{l=1}^L$ from $\mathcal{W}$
- 16 $\mathcal{R} = \emptyset, \mathcal{A}_{ep} = \emptyset$
- 17 **foreach** $B_l \in \mathcal{B}$ **do**
- 18 Agent selects samples from B_l to generate $\mathcal{A}_l$
- 19 $\mathcal{D}_l^{tr} \leftarrow \mathcal{D}^{tr} \cup \mathcal{A}_l$
- 20 Train prototypical network $g_l(\cdot)$ on $\mathcal{D}_l^{tr}$
- 21 Apply $g_l(\cdot)$ on $\mathcal{D}^{val}$ to derive $(F1_l, MacroF1_l, Pr_l)$
- 22 Compute the reward r_l based on Eq. (3)
- 23 $\mathcal{R} = \mathcal{R} \cup \{r_l\}$
- 24 Update the policy network π_θ based on Eq. (5)
- 25 Update the critic network V_ϕ based on Eq. (8)
- 26 $l \leftarrow \arg \max \mathcal{R}$
- 27 **if** $F1_l > F1_0, MacroF1_l > MacroF1_0$, and $Pr_l > Pr_0$ **then**
- 28 $\mathcal{A}_{ep} \leftarrow \mathcal{A}_l$
- 29 **return** $\mathcal{A}_{ep}$

reliable weakly labeled samples from $\mathcal{W}$ to augment the training set. We first apply the current prototypical network $g_0(\cdot)$ on the validation set $\mathcal{D}^{val}$ to get F1, macro-F1 scores, and sample prediction probabilities $(F1_0, MacroF1_0, Pr_0)$ as the baseline performance (Line 14). We then derive a set of sample batches $\mathcal{B} = \{B_l\}_{l=1}^L$ from $\mathcal{W}$.

Then, for each batch B_l , we select samples based on the prediction of the policy network $p(a_j|s_j) = \pi_\theta(s_j)$ and generate $\mathcal{A}_l$ with selected samples. After getting the augmented training set $\mathcal{D}_l^{tr}$ by combining $\mathcal{D}^{tr}$ with $\mathcal{A}_l$ (Line 19), we train the prototypical network $g_l(\cdot)$ on $\mathcal{D}_l^{tr}$ again (Line 20) and apply $g_l(\cdot)$ on $\mathcal{D}^{val}$ to derive F1, macro-F1 scores, and sample prediction probabilities $(F1_l, MacroF1_l, Pr_l)$, (Line 21). The reward is then calculated based on the difference between $(F1_l, MacroF1_l, Pr_l)$ and $(F1_0, MacroF1_0, Pr_0)$. We further update the policy network π_θ and the critic network V_ϕ (Lines 24–25).

After going through all batches, we get a set of rewards $\mathcal{R} = \{r_l\}_{l=0}^L$. We only select $\mathcal{A}_l$ leading to the maximum reward r_l in $\mathcal{R}$ (Line 26) and also ensure that the prototypical network $g_l(\cdot)$ trained on $\mathcal{D}_l^{tr}$ can get better performance in terms of F1, macro-F1 scores, and sample prediction probabilities compared with the original one $g_0(\cdot)$ (Lines 27–28). Then, we consider $\mathcal{A}_l$ as the augmentation set $\mathcal{A}_{ep}$ at the current episode (Line 29).

Based on Algorithm 1, the algorithm runs T iterations, and in each iteration, the data selection process goes through L subsets and trains the prototypical network $g_l(\cdot)$. Therefore,

the time complexity of the algorithm $O(T \cdot L \cdot g)$, where g indicates the time complexity for training the prototypical network.

Warm-up phase. In the early stage of training, the data selection agent takes actions based on randomly initialized parameters. The selected samples could include misclassified instances due to the uncertainty of the policy network. Then, $\mathcal{D}^{tr}$ could contain mislabeled samples, which could damage the performance of FADScr in the long run. Hence, we set up a warm-up phase for the training. After the warm-up phase, all the samples selected during the warm-up phase will be removed from $\mathcal{D}$ and added back to $\mathcal{U}$. The agent will select samples from scratch again. We expect the performance of the data selection agent will become stable with the convergence of parameters during the warm-up phase and can select reliable weakly labeled samples after that. In our experiments, the warm-up phase is 10 episodes.

4 Experiments

4.1 Experimental setup

Datasets. We apply FADScr to detect and classify anomalies on the following three datasets.

- **UNSW-NB15** [39]. The UNSW-NB15 dataset was generated by the IXIA Perfect-Storm tool in the Cyber Range Lab of UNSW Canberra. It consists of normal and five synthetic attacks, including suspension caused by feeding random data (Fuzzers), denial-of-services by overloading the server or network (DoS), attacks with known security problems (Exploits), attack block-ciphers without information about the structure (Generic), and attacks for collecting information (Reconnaissance).
- **IDS2018** [52]. IDS2018 contains detailed network traffic and log files of five victim organizations with 420 PCs and 30 servers. It consists of normal network traffic and three different types of attacking traffic, including automatically synthesized requests of upload, download, screenshots, and keylogging (Bot), denial-of-service by overloading the server or network (DoS), and password crack by brute-force (Bruteforce).
- **CERT** [17]. CERT is an insider threat dataset that contains a collection of synthetic normal and malicious user activities. We use CERT V5.2, which consists of four types of insider threats, including unauthorized uploading data (DataUploading), using a thumb drive to steal data (DataStealing), sending out mass emails causing panic (MassEmail), and unauthorized logging into other computers (UnauthdLogin). Due to the extremely small number of insiders, the number of samples for each anomaly class is small.

Data preprocessing. Both UNSW-NB15 and IDS2018 are multidimensional datasets, where each instance consists of multiple features to describe the basic information. After typical preprocessing steps, each sample in UNSW-NB15 consists of 293 features, while each sample in IDS2018 consists of 77 features. We randomly select 100 normal samples as $\mathcal{L}_N$ and 10 samples from each anomaly class as $\mathcal{L}_A$. To construct an unlabeled dataset $\mathcal{U}$, we randomly select 30,000 normal samples and 3,000 abnormal samples from each anomaly class. The testing dataset contains 20,000 normal samples and 2,000 abnormal samples from each anomaly class.

CERT is a sequential dataset, where each entry indicates one user activity. For CERT V5.2, there are 23 different types of activities, such as logon, logoff, web visiting, and file editing. We group user activities into sessions as data samples, and each session consists of all user activities between logon and logoff operations on a computer. Similarly, we randomly select

Table 1 Statistics of datasets in experiments

Dataset	Class	Training		Testing
		$\mathcal{L}$	$\mathcal{U}$	
UNSW-NB15 (293 features)	Normal	100	30,000	20,000
	Fuzzers	10	3000	2000
	DoS	10	3000	2000
	Exploits	10	3000	2000
	Generic	10	3000	2000
	Reconnaissance	10	3000	2000
IDS2018 (77 features)	Normal	100	30,000	20,000
	Bot	10	3000	2000
	DoS	10	3000	2000
	Bruteforce	10	3000	2000
CERT (various length)	Normal	100	2000	2000
	DataUploading	10	27	28
	DataStealing	10	201	201
	MassEmail	10	10	10
	UnauthdLogin	10	21	21

100 normal samples as $\mathcal{L}_N$ and 10 samples from each anomaly class as $\mathcal{L}_A$. The unlabeled dataset $\mathcal{U}$ is composed of randomly selected 2,000 normal sessions and half of the remaining abnormal sessions (excluding the 10 sessions used in $\mathcal{L}_A$) from each anomaly class. The testing dataset consists of 2,000 normal sessions and the remaining half of the data in each anomalous class. Table 1 summarizes the statistics of datasets used in our experiments.

Baselines. We use two sets of baselines to evaluate the performance of FADScr for anomaly detection and classification, respectively.

Baselines for anomaly detection. We adopt the following six baselines to evaluate the performance of anomaly detection.

- *One-Class SVM (OCSVM)* [51] is a one-class classification model that can detect outliers based on normal samples.
- *Isolation Forest (iForest)* [36] is a widely used tree-based anomaly detection model.
- *Label Random PU Learning (LRPU)* [16] is a classical positive-unlabeled learning approach. It trains a PU learner to predict probabilities that differ by only a constant factor from the true conditional probabilities of being positive.
- *PU learning with a Selection Bias (PUSB)* [28] is an advanced positive-unlabeled learning approach. It tackles the challenge when positive samples include noise.
- *Deep Support Vector Data Description (DeepSVDD)* [46] is a deep learning-based one-class anomaly detection method. The method deploys a neural network to map normal data into a hypersphere with minimum volume. Samples that fall outside of the hypersphere are labeled as anomalies.
- *Deep Semi-Supervised Anomaly Detection (DeepSAD)* [47] is a semi-supervised anomaly detection model. DeepSAD utilizes a small number of labeled normal and anomalous samples as well as an unlabeled dataset to define a hypersphere that can efficiently cover most of the normal samples and further identify the anomalies outside the hypersphere.

- *Deep Isolation Forest for Anomaly Detection (DIF)* [60] is an anomaly detection method that extends Isolation Forest by leveraging deep representations to enhance the separation of normal and anomalous instances.
- *Empirical Cumulative distribution-based Outlier Detection (ECOD)* [35] is a nonparametric anomaly detection method that identifies outliers by analyzing deviations in empirical cumulative distribution functions across multiple dimensions.
- *Learnable Unified Neighborhood-based Anomaly Ranking (LUNAR)* [19] is a graph-based anomaly detection framework that integrates multiple local outlier detection methods using graph neural networks (GNNs) to enhance anomaly detection performance.

OCSVM, iForest, DeepSVDD, DIF, ECOD, and LUNAR as one-class models are trained on the normal set $\mathcal{L}_N$. LRPU and PUBS as positive-unlabeled models are trained on the anomalous set $\mathcal{L}_A$ and unlabeled set $\mathcal{U}$. Deep SAD as a semi-supervised model is trained on the labeled set $\mathcal{L}$ and unlabeled set $\mathcal{U}$.

Baselines for anomaly classification. We adopt the following two baselines to evaluate the performance of anomaly classification.

- *Support Vector Machine (SVM)* [13] is a classification method. We adopt the one-vs-rest strategy for multiclass anomaly detection.
- *Multilayer Perceptron (MLP)* [48] is a classical neural network for multiclass classification.

We train both baselines on the labeled set $\mathcal{L}$ and also adopt the synthetic minority oversampling technique (SMOTE) [9] for oversampling the data in anomaly classes.

Evaluation metrics. For anomaly detection, we consider all anomaly classes as one anomaly class and adopt the Area Under Precision–Recall Curve (AUC-PR), the Area Under Receiver Operating Characteristic Curve (AUC-ROC), and False Positive Rate (FPR) at 95% True Positive Rate (TPR) to measure the performance of FADScr. Especially, FPR at 95% TPR can be considered as the probability that an anomaly is misclassified as normal when the true positive rate is 95%. For anomaly classification, precision, recall, and F1 scores are used to evaluate the performance of FADScr on each abnormal class. We also derive the macro-F1 score to evaluate the performance over all classes. In our experiments, we report results averaged over 20 runs. In the tables of the experimental results section, boldface indicates the models that achieve the best performance. To assess statistical significance, we apply a paired t-test comparing FADScr against the strongest baseline.

4.2 Implementation details

For UNSW-NB15 and IDS2018 datasets, because the input samples are multidimensional data, we adopt a feedforward neural network with three fully connected layers as the implementation of prototypical network $g(\cdot)$, where the dimension of output embedding is 128. For the CERT dataset, because the input samples are sequential data about user activities in a session, we adopt a Gated Recurrent Unit (GRU) neural network [10] to map sequences into an embedding space with a dimension of 64. For all three datasets, as we only have 10 labeled samples in each anomaly class in the initial training set $\mathcal{L}_A$, we initially assign six samples to $\mathcal{D}^{tr}$ to train the prototypical network with 3 samples as the support set $\mathcal{S}$ and 3 samples as the query set $\mathcal{Q}$ and the remaining four samples to $\mathcal{D}^{val}$ for evaluation. In the experiments, we set α and β in Equation 3 as $\alpha = 1$ and $\beta = 0.01$.

In the implementation of actor-critic reinforcement learning, both actor and critic are three-layer feedforward neural networks. In the training stage, we set the number of training

episodes $T = 30$, while in each episode, we generate a set of sample batches $\{B_l\}_{l=1}^L$. For each batch B_l , we select samples from each predicted class in $\mathcal{W}$ based on their closeness to the prototype representations. Especially, for UNSW-NB15 and IDS2018 datasets, in each batch, we select the top 10 samples that are close to the corresponding prototype representation from each class as well as 50 samples that are not close to any prototype representations as hard samples, leading to the sizes of a batch are $|B_l| = 110$ for UNSW-NB15 and $|B_l| = 90$ for IDS2018, respectively. For the CERT dataset, we select the closest sample from each class and five hard samples, leading to the batch size $|B_l| = 10$. The **code and datasets** used in the experiments are available online.¹

4.3 Experimental results

Anomaly detection. First, we evaluate the performance of FADScr in anomaly detection. In particular, after the prototypical network predicts each anomaly into a specific anomaly class, we consider all detected samples in the anomaly classes as anomalies. We compare FADScr for anomaly detection with two traditional one-class classification models (OCSVM and iForest), two positive-unlabeled learning models (LRPU and PUBS), and two advanced deep anomaly detection models (DeepSVDD and DeepSAD). As shown in Table 2, FADScr achieves better performance than all baselines with a large margin on all three datasets. Although DeepSAD, as a semi-supervised model, also uses both labeled and unlabeled datasets, FADScr significantly outperforms DeepSAD in most cases and has a much smaller standard deviation. This means that actively identifying potential anomalies in the unlabeled dataset can improve anomaly detection performance. Meanwhile, DeepSAD still outperforms other baselines, which shows the advantage of using a small set of labeled samples for anomaly detection. LRPU as a PU learner achieves better performance than three one-class models (OCSVM, iForest, and DeepSVDD) on IDS2018 and CERT datasets, meaning that in some cases, leveraging the unlabeled set can improve the performance. However, due to the limited anomalies (positive samples), both LRPU and PUBS underperform FADScr.

Anomaly classification. FADScr can achieve fine-grained anomaly classification based on the prototypical networks, so we further evaluate FADScr for the anomaly classification task. We compare FADScr with two classification models, SVM and MLP. Table 3 shows the results of anomaly classification. In short, FADScr achieves the highest macro-F1 score over all classes, meaning that FADScr can achieve the best performance for anomaly classification. Meanwhile, on the UNSW-NB15 and IDS2018 datasets, FADScr achieves the best performance in most classes in terms of F1 score compared with SVM and MLP. CERT is the most challenging dataset because of extremely limited labeled samples. On the CERT dataset, SVM predicts all anomalies as normal, while MLP has a low recall in the normal set, indicating a high misclassification rate for normal samples. On the other hand, FADScr achieves a significant improvement compared with baselines on CERT. It indicates that for the challenging anomaly detection task with extremely limited samples, leveraging anomalies in the unlabeled dataset is critical to boosting performance.

FADScr with and without data selection. We evaluate the performance improvement of FADScr over the simple prototypical networks without reinforced data selection. We consider three settings as baselines. First, we train a prototypical network on the initially labeled dataset, denoted as *ProtoNet*. Second, after we train the prototypical network on the initial dataset, we apply the network to label the unlabeled dataset $\mathcal{U}$ and use the whole weakly labeled

¹ <https://github.com/hanxiao0607/FADScr>.

Table 2 Results on anomaly detection (mean $\pm$ std.)

		UNSW-NB15	IDS2018	CERT
AUC-PR $\uparrow$	OCSVM	0.4999 $\pm$ 0.0029	0.3069 $\pm$ 0.0115	0.1161 $\pm$ 0.0170
	iForest	0.7337 $\pm$ 0.0537	0.2914 $\pm$ 0.0151	0.1321 $\pm$ 0.0198
	LRPU	0.6559 $\pm$ 0.0084	0.4365 $\pm$ 0.0444	0.1380 $\pm$ 0.0520
	PUSB	0.6119 $\pm$ 0.0246	0.3774 $\pm$ 0.0158	0.1147 $\pm$ 0.0000
	DeepSVDD	0.6088 $\pm$ 0.0359	0.4704 $\pm$ 0.1126	0.1192 $\pm$ 0.0068
	DeepSAD	0.8103 $\pm$ 0.1958	0.5945 $\pm$ 0.1064	0.1304 $\pm$ 0.0176
	DIF	0.5106 $\pm$ 0.0615	0.4921 $\pm$ 0.0550	0.1552 $\pm$ 0.0172
	ECOD	0.5167 $\pm$ 0.0044	0.1669 $\pm$ 0.0006	0.1285 $\pm$ 0.0143
	LUNAR	0.7983 $\pm$ 0.0600	0.5107 $\pm$ 0.0532	0.1615 $\pm$ 0.0325
AUC-ROC $\uparrow$	FADScR	0.9249 $\pm$ 0.0206*	0.7121 $\pm$ 0.0592**	0.2531 $\pm$ 0.0397**
	OCSVM	0.7499 $\pm$ 0.0029	0.6517 $\pm$ 0.0187	0.5671 $\pm$ 0.0436
	iForest	0.8991 $\pm$ 0.0315	0.5489 $\pm$ 0.0280	0.5937 $\pm$ 0.0572
	LRPU	0.7420 $\pm$ 0.0063	0.6338 $\pm$ 0.0289	0.5432 $\pm$ 0.0482
	PUSB	0.7089 $\pm$ 0.0184	0.5953 $\pm$ 0.0103	0.5000 $\pm$ 0.0000
	DeepSVDD	0.7436 $\pm$ 0.0316	0.7128 $\pm$ 0.0905	0.5279 $\pm$ 0.0204
	DeepSAD	0.9024 $\pm$ 0.1497	0.7840 $\pm$ 0.0896	0.5601 $\pm$ 0.0558
	DIF	0.7132 $\pm$ 0.0459	0.7933 $\pm$ 0.0353	0.6044 $\pm$ 0.0481
	ECOD	0.7630 $\pm$ 0.0039	0.2951 $\pm$ 0.0022	0.5705 $\pm$ 0.0659
FPR (95% TPR) $\downarrow$	LUNAR	0.9374 $\pm$ 0.0257	0.7557 $\pm$ 0.0474	0.6502 $\pm$ 0.0652
	FADScR	0.9783 $\pm$ 0.0066*	0.9299 $\pm$ 0.0219**	0.7406 $\pm$ 0.0357**
	OCSVM	0.9941 $\pm$ 0.0007	0.9149 $\pm$ 0.0140	0.9138 $\pm$ 0.0483
	iForest	0.2093 $\pm$ 0.0660	0.9549 $\pm$ 0.0452	0.7526 $\pm$ 0.1303
	LRPU	0.3738 $\pm$ 0.0675	0.5236 $\pm$ 0.1379	0.8827 $\pm$ 0.0643
	PUSB	0.9842 $\pm$ 0.0328	0.8691 $\pm$ 0.1974	0.9522 $\pm$ 0.0107
	DeepSVDD	0.6664 $\pm$ 0.1704	0.7159 $\pm$ 0.2406	0.8619 $\pm$ 0.0678
	DeepSAD	0.2222 $\pm$ 0.3990	0.4326 $\pm$ 0.3343	0.6997 $\pm$ 0.0938
	DIF	0.7464 $\pm$ 0.0870	0.5528 $\pm$ 0.1405	0.7562 $\pm$ 0.1253
	ECOD	0.5333 $\pm$ 0.0086	0.9791 $\pm$ 0.0022	0.7807 $\pm$ 0.1093
	LUNAR	0.4509 $\pm$ 0.2710	0.7717 $\pm$ 0.1938	0.7454 $\pm$ 0.1565
	FADScR	0.0333 $\pm$ 0.0133*	0.1659 $\pm$ 0.0986**	0.6849 $\pm$ 0.0854

$\uparrow$ indicates larger value is better; $\downarrow$ indicates lower value is better

Significantly outperforms DeepSAD at the: * 0.05 and ** 0.01 level, paired *t*-test

dataset $\mathcal{W}$ as the augmentation set ($\mathcal{A} \leftarrow \mathcal{W}$) to retrain the prototypical network. We evaluate the performance of the prototypical network after retraining on the test set, denoted as *ProtoNet**. The third one is similar to *ProtoNet**, but we retrain the prototypical network based on samples in $\mathcal{W}$ having top 10% of the highest probabilities in each class, which means high confidence belonging to a class, denoted as *ProtoNet* (10%)*.

We also compare FADScR with our previous work *FADS* [21], which does not have the reward for choosing the hard samples for performance improvement. We compare four models for anomaly detection and classification. For the anomaly classification, we report the macro-precision, recall, and F1 score here.

Table 3 Results on anomaly classification (mean $\pm$ std.)

Dataset	Class	Precision			Recall			F1		
		SVM	MLP	FADScr	SVM	MLP	FADScr	SVM	MLP	FADScr
UNSW-NB15	Normal	0.8135 $\pm$ 0.0089	0.8399 $\pm$ 0.0099	0.9979 $\pm$ 0.0038	0.9873 $\pm$ 0.0406	0.9874 $\pm$ 0.0358	0.9607 $\pm$ 0.0129	0.8957 $\pm$ 0.0055	0.9110 $\pm$ 0.0056	0.9789 $\pm$ 0.0066
	Fuzzers	0.3858 $\pm$ 0.0791	0.3538 $\pm$ 0.0720	0.3824 $\pm$ 0.0469	0.1121 $\pm$ 0.0790	0.1244 $\pm$ 0.0476	0.4335 $\pm$ 0.0962	0.1502 $\pm$ 0.0268	0.1726 $\pm$ 0.0310	0.4036 $\pm$ 0.0635
	DoS	0.5341 $\pm$ 0.0389	0.5002 $\pm$ 0.0525	0.4602 $\pm$ 0.0726	0.3729 $\pm$ 0.1242	0.3335 $\pm$ 0.1077	0.4532 $\pm$ 0.0917	0.4302 $\pm$ 0.0933	0.3884 $\pm$ 0.0839	0.4539 $\pm$ 0.0744
	Exploits	0.3946 $\pm$ 0.0936	0.4224 $\pm$ 0.0748	0.3749 $\pm$ 0.0650	0.1862 $\pm$ 0.0849	0.2513 $\pm$ 0.0635	0.3900 $\pm$ 0.0769	0.2336 $\pm$ 0.0432	0.3048 $\pm$ 0.0599	0.3782 $\pm$ 0.0591
	Generic	0.9762 $\pm$ 0.0197	0.9579 $\pm$ 0.0247	0.9217 $\pm$ 0.0912	0.9666 $\pm$ 0.0253	0.9662 $\pm$ 0.0161	0.9649 $\pm$ 0.0389	0.9710 $\pm$ 0.0153	0.9618 $\pm$ 0.0148	0.9414 $\pm$ 0.0640
IDS2018	Reconnaissance	0.3561 $\pm$ 0.1102	0.3865 $\pm$ 0.0864	0.3212 $\pm$ 0.0614	0.1392 $\pm$ 0.0976	0.2258 $\pm$ 0.0751	0.3631 $\pm$ 0.0710	0.1740 $\pm$ 0.0404	0.2696 $\pm$ 0.0497	0.3391 $\pm$ 0.0604
	macro-average	0.5767 $\pm$ 0.0281	0.5768 $\pm$ 0.0251	0.5764 $\pm$ 0.0290	0.4607 $\pm$ 0.0463	0.4815 $\pm$ 0.0258	0.5942 $\pm$ 0.0303	0.4758 $\pm$ 0.0199	0.5014 $\pm$ 0.0190	0.5825 $\pm$ 0.0288**
	Normal	0.9904 $\pm$ 0.0180	0.9945 $\pm$ 0.0098	0.9923 $\pm$ 0.0172	0.5657 $\pm$ 0.1111	0.8392 $\pm$ 0.0363	0.8853 $\pm$ 0.0391	0.6941 $\pm$ 0.0454	0.9019 $\pm$ 0.0293	0.9350 $\pm$ 0.0164
	Bot	0.6482 $\pm$ 0.2551	0.5348 $\pm$ 0.1855	0.5134 $\pm$ 0.1452	0.7578 $\pm$ 0.2499	0.9477 $\pm$ 0.1526	0.9711 $\pm$ 0.1110	0.6311 $\pm$ 0.0961	0.6538 $\pm$ 0.0846	0.6513 $\pm$ 0.0635
	DoS	0.1934 $\pm$ 0.0456	0.6615 $\pm$ 0.1765	0.8520 $\pm$ 0.1090	0.6914 $\pm$ 0.1263	0.7047 $\pm$ 0.0997	0.8136 $\pm$ 0.0579	0.3201 $\pm$ 0.1082	0.6796 $\pm$ 0.1179	0.8258 $\pm$ 0.0429
CERT	Bruteforce	0.4814 $\pm$ 0.0965	0.6722 $\pm$ 0.0773	0.7807 $\pm$ 0.0779	0.9299 $\pm$ 0.1292	0.9629 $\pm$ 0.0886	0.8444 $\pm$ 0.1376	0.6452 $\pm$ 0.1215	0.8018 $\pm$ 0.0797	0.8062 $\pm$ 0.0903
	Macro-average	0.5784 $\pm$ 0.0517	0.7157 $\pm$ 0.0217	0.7846 $\pm$ 0.0558	0.7362 $\pm$ 0.0759	0.8636 $\pm$ 0.0515	0.8786 $\pm$ 0.0436	0.5726 $\pm$ 0.0659	0.7593 $\pm$ 0.0452	0.8046 $\pm$ 0.0403**
	Normal	0.9599 $\pm$ 0.0188	0.9016 $\pm$ 0.0032	0.9354 $\pm$ 0.0108	0.4883 $\pm$ 0.2146	0.9464 $\pm$ 0.0207	0.8874 $\pm$ 0.0215	0.6205 $\pm$ 0.1991	0.9243 $\pm$ 0.0102	0.9097 $\pm$ 0.0097
	DataUploading	0.2076 $\pm$ 0.1397	0.3474 $\pm$ 0.1479	0.4684 $\pm$ 0.1665	0.8722 $\pm$ 0.0911	0.7222 $\pm$ 0.1464	0.8125 $\pm$ 0.1094	0.3161 $\pm$ 0.1713	0.4503 $\pm$ 0.1174	0.5787 $\pm$ 0.1128
	DataStealing	0.3283 $\pm$ 0.1198	0.0506 $\pm$ 0.1264	0.4044 $\pm$ 0.0525	0.7124 $\pm$ 0.1206	0.0100 $\pm$ 0.0306	0.4042 $\pm$ 0.1173	0.4374 $\pm$ 0.1264	0.0159 $\pm$ 0.0480	0.3983 $\pm$ 0.0725
UnauthdLogin	MassEmail	0.0223 $\pm$ 0.0108	0.1436 $\pm$ 0.2089	0.1111 $\pm$ 0.0546	0.6450 $\pm$ 0.2417	0.4250 $\pm$ 0.2468	0.6418 $\pm$ 0.1638	0.0428 $\pm$ 0.0202	0.1949 $\pm$ 0.1669	0.1813 $\pm$ 0.0724
	UnauthdLogin	0.2152 $\pm$ 0.1901	0.3456 $\pm$ 0.2120	0.5245 $\pm$ 0.1700	0.6929 $\pm$ 0.1758	0.5143 $\pm$ 0.1222	0.7524 $\pm$ 0.0935	0.2987 $\pm$ 0.2198	0.3753 $\pm$ 0.1337	0.6010 $\pm$ 0.1259
	Macro-average	0.3467 $\pm$ 0.0784	0.3578 $\pm$ 0.0629	0.4888 $\pm$ 0.0419	0.6822 $\pm$ 0.1017	0.5236 $\pm$ 0.0521	0.6997 $\pm$ 0.0550	0.3432 $\pm$ 0.1205	0.3922 $\pm$ 0.0468	0.5338 $\pm$ 0.0388**

Significantly outperforms MLP in terms of macro-F1 at the: * 0.05 and ** 0.01 level, paired *t*-test

Table 4 Performance of FADScr with or without reinforced data selection (mean $\pm$ std.)

Dataset	Approach	Anomaly detection		Anomaly classification			
		AUC-PR↑	AUC-ROC↑	FPR at 95% TPR↓	Precision	Recall	F1
UNSW-NB15	ProtoNet	0.8522 ± 0.0424	0.9504 ± 0.0155	0.0873 ± 0.0405	0.5180 ± 0.0301	0.5503 ± 0.0359	0.5205 ± 0.0322
	ProtoNet*	0.7245 ± 0.0988	0.8865 ± 0.0796	0.1821 ± 0.1649	0.4153 ± 0.0556	0.4673 ± 0.0713	0.4262 ± 0.0634
	ProtoNet*(10%)	0.7111 ± 0.0685	0.8908 ± 0.0374	0.2153 ± 0.0862	0.4595 ± 0.0420	0.5194 ± 0.0517	0.4645 ± 0.0467
	FADS	0.9050 ± 0.0232	0.9693 ± 0.0102	0.0398 ± 0.0121	0.5357 ± 0.0841	0.5813 ± 0.0296	0.5676 ± 0.0302
	FADScr	0.9249 ± 0.0206**	0.9783 ± 0.0066**	0.0333 ± 0.0133*	0.5764 ± 0.0290	0.5942 ± 0.0303*	0.5825 ± 0.0288**
IDS2018	ProtoNet	0.6894 ± 0.0433	0.9180 ± 0.0276	0.2758 ± 0.1406	0.7104 ± 0.0595	0.8396 ± 0.0588	0.7388 ± 0.0415
	ProtoNet*	0.5081 ± 0.0722	0.8208 ± 0.0669	0.4142 ± 0.2097	0.5329 ± 0.0819	0.6317 ± 0.1166	0.5471 ± 0.0767
	ProtoNet*(10%)	0.5288 ± 0.0670	0.8555 ± 0.0386	0.3360 ± 0.1740	0.6371 ± 0.0776	0.7620 ± 0.0808	0.6400 ± 0.0627
	FADS	0.6780 ± 0.0343	0.9219 ± 0.0231	0.2835 ± 0.1209	0.7657 ± 0.0403	0.8800 ± 0.0385	0.7902 ± 0.0275
	FADScr	0.7121 ± 0.0592**	0.9299 ± 0.0219**	0.1659 ± 0.0986**	0.7846 ± 0.0558*	0.8786 ± 0.0436	0.8046 ± 0.0403
CERT	ProtoNet	0.1784 ± 0.0569	0.6314 ± 0.0729	0.9777 ± 0.0637	0.3474 ± 0.0499	0.5501 ± 0.0724	0.3613 ± 0.0697
	ProtoNet*	0.1788 ± 0.0305	0.6738 ± 0.0564	0.7316 ± 0.1062	0.3418 ± 0.0443	0.6141 ± 0.1017	0.3551 ± 0.0632
	ProtoNet*(10%)	0.2367 ± 0.0347	0.7066 ± 0.0404	0.7651 ± 0.1248	0.3847 ± 0.0465	0.7160 ± 0.0725	0.4353 ± 0.0627
	FADS	0.2397 ± 0.0383	0.7161 ± 0.0451	0.7083 ± 0.1322	0.4119 ± 0.0379	0.7243 ± 0.0453	0.4694 ± 0.0449
	FADScr	0.2531 ± 0.0397	0.7406 ± 0.0357*	0.6849 ± 0.0854	0.4888 ± 0.0419**	0.6997 ± 0.0550	0.5338 ± 0.0388**

Significantly outperforms FADS at the: * 0.05 and ** 0.01 level, paired t-test

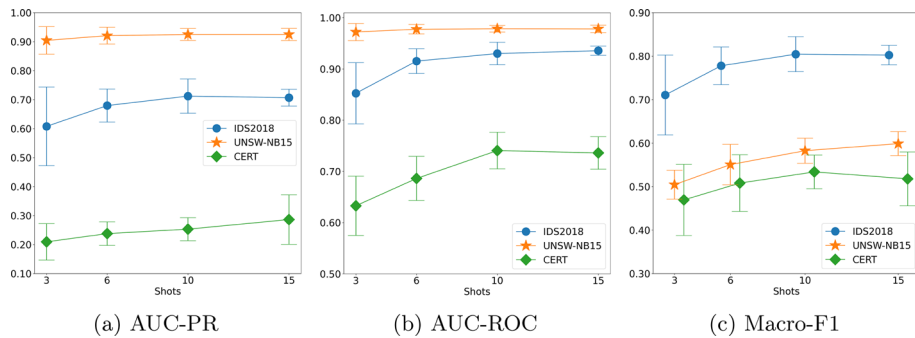


Fig. 3 Performance of FADScR with different numbers of initially labeled anomalies in one class. **a**, **b** for anomaly detection and **c** for anomaly classification

Table 4 shows that FADScR achieves unanimously better performances over baselines on three datasets for both anomaly detection and classification tasks. Meanwhile, we can notice that, in general, both ProtoNet* and ProtoNet* (10%) cannot beat regular ProtoNet, which shows that simply using weakly labeled samples for training will hurt the performance due to incorrect labels. Furthermore, FADScR still has the largest performance gains on the CERT dataset in terms of macro-F1 for classification, which shows the criticalness of augmenting the training set with reliable samples in the case of limited samples. When comparing the performance of FADS and FADScR, we can notice that FADScR with a combinatorial reward can further improve the performance in terms of AUC-PR by a large margin and also achieve much better F1 scores, especially on the CERT and UNSW-NB15 datasets. Another advantage of FADScR is that, in most cases, it can have a lower standard deviation compared with all other baselines.

Performance of FADScR with Different Numbers of Initially Labeled Anomalies. Table 4 indicates that the reinforced data selection strategy for training data augmentation can improve the performance of anomaly detection and classification. We then evaluate the performance under the settings where different numbers of labeled anomalies are available at the beginning of the training phase. Figure 3 shows the experimental results in terms of AUC-PR, AUC-ROC (anomaly detection), and macro-F1 (anomaly classification). We can notice that on IDS2018 and UNSW-NB15 datasets, with more labeled samples available, the performance of FADScR keeps improving. Furthermore, the standard deviations also decrease with more available samples on IDS2018 and UNSW-NB15. This means that with more anomalies available initially, the performance of anomaly detection and classification will become more stable. The performance of anomaly detection on the CERT dataset also basically follows a similar trend, i.e., more available anomalies leading to better performance. On the other hand, the macro-F1 scores for anomaly classification remain stable with more samples. This could be because the fine-grained classification task on CERT requires more anomalies to improve the performance further due to its difficulty.

Performance gains of FADScR over ProtoNet with different numbers of initially labeled anomalies. Figure 4 shows that, on IDS2018 and UNSW-NB15 datasets, the performance gains compared with ProtoNet without reinforced data selection are decreasing when more labeled samples are available. This indicates that once prototypical networks have sufficient samples for training, incorporating more samples will not change the results significantly. However, different from the results on UNSW-NB15 and IDS2018, the performance gains

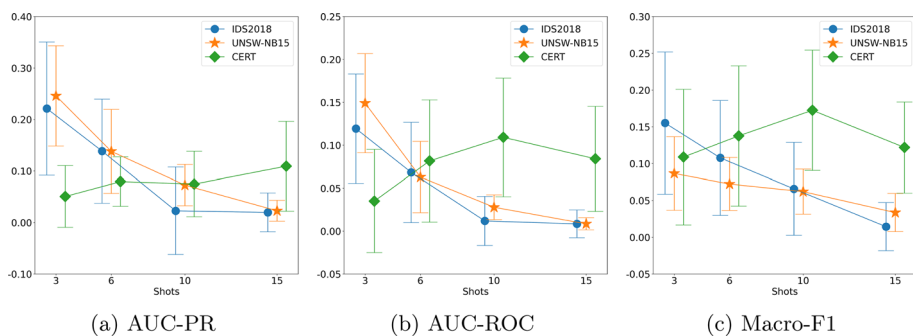


Fig. 4 Performance gains of FADScr over ProtoNet with different numbers of initially labeled anomalies

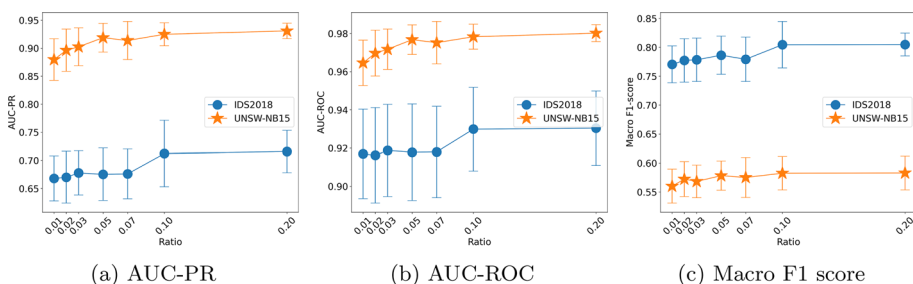


Fig. 5 Performance of FADScr with various anomaly ratios in the unlabeled set

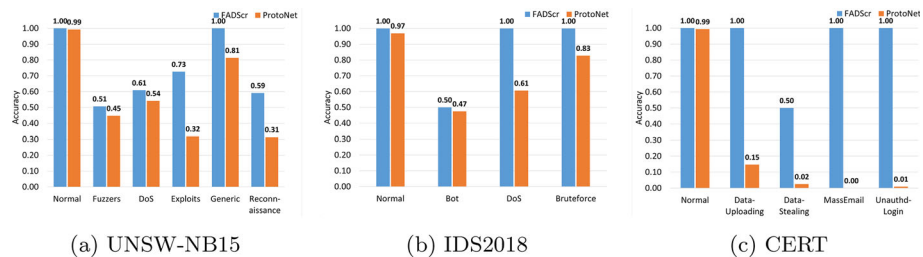
on CERT increase when the number of available anomalies in each class increases. It could be because three samples are not sufficient to train prototypical networks for insider threat detection due to the difficulty. As a result, the weakly labeled samples are full of noise, and no correctly labeled samples are available for retraining. Meanwhile, we can observe that FADScr consistently achieves positive gains over ProtoNet.

Performance of FADScr with Various Anomaly Ratios in the Unlabeled Dataset. For the UNSW-NB15 and IDS2018 datasets, we set the ratio of anomalies in the unlabeled dataset $\mathcal{U}$ as 0.1 by default. We further tune this ratio to check the performance change of FADScr. Note that because of limited labeled anomalies, we do not conduct this experiment on CERT. Figure 5 shows the results of different ratios of anomalies in $\mathcal{U}$ from 0.01 to 0.2. First, we can observe that with more anomalies in $\mathcal{U}$, the performance of FADScr on anomaly detection and classification keeps improving. Especially, for the IDS2018 dataset, in terms of AUC-PR and macro-F1 score, the performance of FADScr keeps improving with more anomalies in the unlabeled dataset. For the UNSW-NB15 dataset, the performance of FADScr is improved when the ratios of anomalies increase from 0.01 to 0.1 and then remains stable even when more anomalies are included in the unlabeled dataset. This could be because for the UNSW-NB15 dataset, the 10% of anomalies already cover the majority of anomaly types, and adding more anomalies in the unlabeled dataset can only reduce the standard deviation.

Ablation study on reward components. The reward function defined in Equation 3 consists of three parts: rewards for anomaly classification and detection as well as the improvement on hard samples. In this experiment, we check the performance of FADScr if we keep only one reward component. Table 5 indicates that incorporating all three reward components can improve performance compared with the model with only one reward component. Meanwhile,

Table 5 Impact of each reward component

Dataset	Reward	AUC-PR	AUC-ROC	Macro-F1
UNSW-NB15	r^d	0.9050 ± 0.0182	0.9585 ± 0.0053	0.5587 ± 0.0293
	r^c	0.9002 ± 0.0259	0.9574 ± 0.0078	0.5622 ± 0.0235
	r^h	0.9025 ± 0.0169	0.9578 ± 0.0049	0.5612 ± 0.0270
	Combine	0.9249 ± 0.0206	0.9783 ± 0.0066	0.5825 ± 0.0288
IDS2018	r^d	0.6714 ± 0.0269	0.9237 ± 0.0140	0.7873 ± 0.0241
	r^c	0.6656 ± 0.0325	0.9216 ± 0.0150	0.7901 ± 0.0282
	r^h	0.6651 ± 0.0355	0.9161 ± 0.0233	0.7874 ± 0.0309
	Combine	0.7121 ± 0.0592	0.9299 ± 0.0219	0.8046 ± 0.0403
CERT	r^d	0.2432 ± 0.0379	0.7111 ± 0.0336	0.4685 ± 0.0540
	r^c	0.2382 ± 0.0338	0.6978 ± 0.0369	0.4707 ± 0.0409
	r^h	0.2456 ± 0.0390	0.7042 ± 0.0446	0.4712 ± 0.0414
	Combine	0.2531 ± 0.0397	0.7406 ± 0.0357	0.5338 ± 0.0388

**Fig. 6** Accuracy of the data selection agent

FADScr with one reward component (shown in Table 5) already achieves better performance than ProtoNet (shown in Table 4).

Accuracy of the reinforced data selection agent. We further evaluate whether the reinforced data selection agent is able to select the high-quality samples from the weakly labeled dataset $\mathcal{W}$. We calculate the accuracy based on all selected samples when the training procedure for data selection is finished. Figure 6 shows the results, where the blue bar indicates the accuracy of the FADScr agent while the orange bar indicates the accuracy of initial prototypical networks. We can notice that the initial prototypical networks cannot achieve good performance on anomaly classification, which means the weakly labeled dataset $\mathcal{W}$ consists of a lot of mis-labeled samples. However, the samples selected from the agent have much higher chances of being correct, indicating that the agent is able to identify high-quality weakly labeled samples from $\mathcal{W}$. Therefore, using the selected samples to augment the training dataset can improve the performance of the few-shot anomaly detection model.

Convergence and running time analysis. We further conduct the convergence analysis by comparing FADScr and FADS. As shown in Fig. 7, FADScr consistently outperforms FADS across all datasets in terms of anomaly detection (AUC-ROC) and classification (macro-F1 scores). The performance improves rapidly in the first two iterations, with FADScr achieving higher final scores compared to FADS. Moreover, FADScr converges faster and reaches a plateau around the second iteration, whereas FADS stabilizes at a lower performance level,

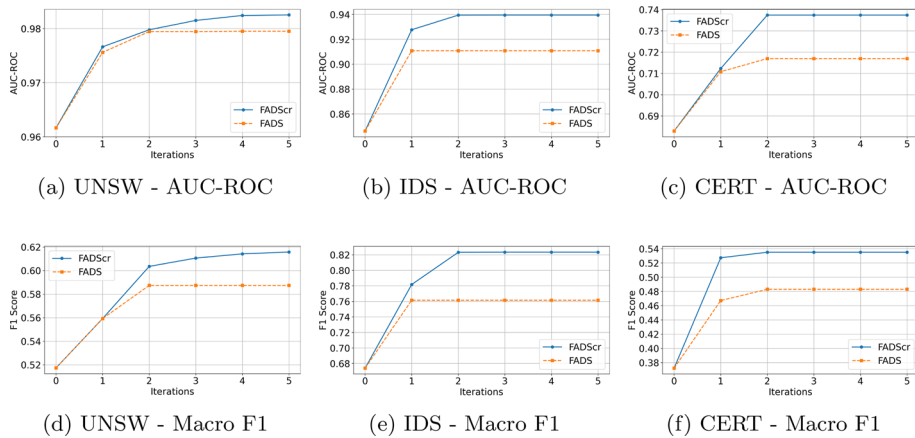


Fig. 7 Performance comparison of FADS and FADScr across different datasets and evaluation metrics. The first row presents anomaly detection results (AUC-ROC), while the second row shows classification performance (Macro-F1 Score)

Table 6 Training and inference time of FADS and FADScr

Dataset	Model	Training time (s/iteration)	Inference time (s)
UNSW-NB15	FADS	94.65 ± 0.66	95.04
	FADScr	144.10 ± 2.17	96.21
IDS2018	FADS	55.82 ± 2.08	19.09
	FADScr	130.84 ± 0.87	19.28
CERT	FADS	1512.80 ± 1.33	5.06
	FADScr	1740.60 ± 2.06	4.98

demonstrating the effectiveness of the FADScr model in optimizing both AUC-ROC and F1 scores more efficiently than FADS.

Table 6 showcases that FADScr delivers substantial gains with only a modest computational premium. On three datasets, the per-iteration training of FADScr is slower than FADS, but it achieves richer data selection that drives its superior detection performance, while inference time remains almost unchanged. These results underscore that FADScr's data selection with combinatorial rewards delivers meaningful accuracy gains while having a negligible impact on inference speed, making the method practical for deployment.

5 Conclusions

In this paper, we introduced FADScr, a few-shot anomaly detection and classification model that leverages reinforced data selection to enhance model performance iteratively. The core idea behind FADScr is to improve a few-shot learning model by systematically selecting potential anomalies from an unlabeled dataset and incorporating them into the training process through a reinforcement learning-based agent. Specifically, the framework first trains a prototypical network with a limited number of labeled samples and then refines it by extracting weakly labeled samples from the unlabeled dataset. A reinforced data selection

agent is trained to identify high-confidence yet informative weakly labeled samples, thereby augmenting the training set in a principled manner.

Improving from our previous work (FADS), FADScr introduces a combinatorial reward mechanism that prioritizes samples with high uncertainty, allowing the model to make more informed updates to the class prototypes. This iterative training process enables FADScr to progressively refine anomaly detection and classification capabilities, even in scenarios with extremely limited labeled data. Extensive experiments demonstrate that FADScr outperforms various baseline methods, including unsupervised, semi-supervised, and few-shot learning-based approaches, validating its effectiveness in both anomaly detection and fine-grained anomaly classification.

For future work, we plan to extend FADScr by exploring new reinforcement learning strategies for anomaly detection, particularly focusing on identifying entirely new classes of anomalies, i.e., open-set anomaly detection. One major challenge in anomaly detection is that real-world anomalies are often diverse and evolving, making it difficult to predefine all possible anomaly categories during training. To address this, we propose leveraging reinforcement learning for new class discovery by integrating an anomaly discovery agent that actively explores the latent structure of unlabeled anomalies and attempts to cluster them into new anomaly categories. In essence, we aim to build a more adaptive anomaly detection framework that can generalize beyond known anomaly types and continuously learn from dynamic, real-world data streams.

Acknowledgements The work is significantly extended from our previous work [21]. In our previous work [21], we proposed FADS for few-shot anomaly detection and classification. In this work, we extend our framework to FADScr by further designing a combinatorial reward to enhance the data selection step from the weakly labeled set. This work was supported in part by NSF Grants 1564250, 2103829, and 2348391.

Declarations

Conflict of interest The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Akcay S, Atapour-Abarghouei A, Breckon TP (2018) GANomaly: Semi-supervised anomaly detection via adversarial training. In: Asian conference on computer vision, Springer, pp 622–637
2. An J, Cho S (2015) Variational autoencoder based anomaly detection using reconstruction probability. *Spec Lect IE* 2(1):1–18
3. Ayyad A, Li Y, Muaz R, et al (2021) Semi-supervised few-shot learning with prototypical random walks. In: AAAI Workshop on meta-learning and MetaDL Challenge. PMLR, pp 45–57
4. Belton N, Hagos MT, Lawlor A, et al (2023) Fewsome: one-class few shot anomaly detection with Siamese networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 2978–2987
5. Cao Y, Wan Q, Shen W et al (2022) Informative knowledge distillation for image anomaly segmentation. *Knowl Based Syst* 248:108846

6. Cao Y, Xu X, Zhang J, et al (2024) A survey on visual anomaly detection: challenge, approach, and prospect. [arXiv:2401.16402](#)
7. Chalapathy R, Chawla S (2019) Deep learning for anomaly detection: a survey. [arXiv:1901.03407](#)
8. Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. *ACM Comput Surv (CSUR)* 41(3):1–58
9. Chawla NV, Bowyer KW, Hall LO et al (2002) Smote: synthetic minority over-sampling technique. *J Artif Intell Res* 16:321–357
10. Cho K, Van Merriënboer B, Bahdanau D, et al (2014) On the properties of neural machine translation: encoder–decoder approaches. [arXiv:1409.1259](#)
11. Ding K, Zhou Q, Tong H, et al (2021) Few-shot network anomaly detection via cross-network meta-learning. In: Proceedings of the web conference 2021, WWW '21, pp 2448–2456
12. Ding K, Wang J, Caverlee J, et al (2022) Meta propagation networks for graph few-shot semi-supervised learning. In: Proceedings of the AAAI conference on artificial intelligence, pp 6524–6531
13. Drucker H, Burges CJ, Kaufman L et al (1997) Support vector regression machines. *Adv Neural Inf Process Syst* 9:155–161
14. Du M, Li F, Zheng G, et al (2017) Deeplog: anomaly detection and diagnosis from system logs through deep learning. In: Proceedings of the 2017 ACM SIGSAC conference on computer and communications security, pp 1285–1298
15. Elaziz EA, Fathalla R, Shaheen M (2023) Deep reinforcement learning for data-efficient weakly supervised business process anomaly detection. *J Big Data* 10(1):33
16. Elkan C, Noto K (2008) Learning classifiers from only positive and unlabeled data. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 213–220
17. Glasser J, Lindauer B (2013) Bridging the gap: a pragmatic approach to generating insider threat data. In: 2013 IEEE security and privacy workshops, pp 98–104
18. Goodfellow I, Pouget-Abadie J, Mirza M et al (2020) Generative adversarial networks. *Commun ACM* 63(11):139–144
19. Goodge A, Hooi B, Ng SK, et al (2022) Lunar: unifying local outlier detection methods via graph neural networks. In: Proceedings of the AAAI conference on artificial intelligence, pp 6737–6745
20. Gu Z, Yang T, Ma L (2023) Prototype-aware contrastive knowledge distillation for few-shot anomaly detection. In: *BMVC*, pp 259–263
21. Han X, Xu D, Yuan S, et al (2022) Few-shot anomaly detection and classification through reinforced data selection. In: 2022 IEEE international conference on data mining (ICDM)
22. He H, Zhang J, Chen H, et al (2024) A diffusion-based framework for multi-class anomaly detection. In: Proceedings of the AAAI conference on artificial intelligence, pp 8472–8480
23. Hendrycks D, Mazeika M, Dietterich T (2018) Deep anomaly detection with outlier exposure. In: International conference on learning representations
24. Hu T, Zhang J, Yi R, et al (2024) Anomalydiffusion: Few-shot anomaly image generation with diffusion model. In: Proceedings of the AAAI conference on artificial intelligence, pp 8526–8534
25. Huang C, Wu Y, Zuo Y, et al (2018) Towards experienced anomaly detector through reinforcement learning. In: Proceedings of the AAAI conference on artificial intelligence
26. Huang C, Guan H, Jiang A, et al (2022) Registration based few-shot anomaly detection. In: European conference on computer vision. Springer, pp 303–319
27. Huang C, Guan H, Jiang A, et al (2024) Few-shot anomaly detection via category-agnostic registration learning. *IEEE Trans Neural Netw Learn Syst*
28. Kato M, Teshima T, Honda J (2018) Learning from positive and unlabeled data with a selection bias. In: International conference on learning representations
29. Kingma DP, Welling M, et al (2019) An introduction to variational autoencoders. *Found Trends® Mach Learn* 12(4):307–392
30. Koizumi Y, Murata S, Harada N, et al (2019) Sniper: Few-shot learning for anomaly detection to minimize false-negative rate with ensured true-positive rate. In: 2019 IEEE international conference on acoustics, speech and signal processing
31. Konda VR, Tsitsiklis JN (2000) Actor-critic algorithms. In: Advances in neural information processing systems, pp 1008–1014
32. Kurt MN, Ogundijo O, Li C et al (2019) Online cyber-attack detection in smart grid: a reinforcement learning approach. *IEEE Trans Smart Grid* 10(5):5174–5185
33. Lee JY, Lee W, Choi J, et al (2023) Few-shot anomaly detection with adversarial loss for robust feature representations. [arXiv:2312.03005](#)
34. Li Y, Goodge A, Liu F, et al (2024) Promptad: zero-shot anomaly detection using text prompts. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp 1093–1102

35. Li Z, Zhao Y, Hu X et al (2022) ECOD: unsupervised outlier detection using empirical cumulative distribution functions. *IEEE Trans Knowl Data Eng* 35(12):12181–12193
36. Liu FT, Ting KM, Zhou ZH (2008) Isolation forest. In: 2008 Eighth IEEE international conference on data mining. IEEE, pp 413–422
37. Lu Y, Yu F, Reddy MKK, et al (2020) Few-shot scene-adaptive anomaly detection. In: European conference on computer vision. Springer, pp 125–141
38. Meng W, Liu Y, Zhu Y, et al (2019) Loganomaly: unsupervised detection of sequential and quantitative anomalies in unstructured logs. In: IJCAI, pp 4739–4745
39. Moustafa N, Slay J (2015) UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: 2015 Military communications and information systems conference (MilCIS). IEEE, pp 1–6
40. Oh Mh, Iyengar G (2019) Sequential anomaly detection using inverse reinforcement learning. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1480–1490
41. Pang G, Shen C, Cao L, et al (2020) Deep learning for anomaly detection: a review. [arXiv:2007.02500](https://arxiv.org/abs/2007.02500)
42. Pang G, Ding C, Shen C, et al (2021a) Explainable deep few-shot anomaly detection with deviation networks. [arXiv:2108.00462](https://arxiv.org/abs/2108.00462)
43. Pang G, van den Hengel A, Shen C, et al (2021b) Toward deep supervised anomaly detection: reinforcement learning from partially labeled anomaly data. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery and data mining, pp 1298–1308
44. Pang G, Shen C, Jin H, et al (2023) Deep weakly-supervised anomaly detection. In: Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining, pp 1795–1807
45. Rudolph M, Wandt B, Rosenhahn B (2021) Same same but different: Semi-supervised defect detection with normalizing flows. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp 1907–1916
46. Ruff L, Vandermeulen R, Goernitz N, et al (2018) Deep one-class classification. In: Proceedings of the 35th international conference on machine learning, pp 4393–4402
47. Ruff L, Vandermeulen RA, Görnitz N, et al (2020) Deep semi-supervised anomaly detection. In: Proceedings of the international conference on learning representations
48. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning Internal Representations by Error Propagation. MIT Press, Cambridge, pp 318–362
49. Schlegl T, Seeßböck P, Waldstein SM, et al (2017) Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In: International conference on information processing in medical imaging. Springer, pp 146–157
50. Schlegl T, Seeßböck P, Waldstein SM et al (2019) f-AnoGAN: fast unsupervised anomaly detection with generative adversarial networks. *Med Image Anal* 54:30–44
51. Schölkopf B, Williamson RC, Smola AJ, et al (1999) Support vector method for novelty detection. In: NIPS. Citeseer, pp 582–588
52. Sharafaldin I, Lashkari AH, Ghorbani AA et al (2018) Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp* 1(2018):108–116
53. Sheynin S, Benaim S, Wolf L (2021) A hierarchical transformation-discriminating generative model for few shot anomaly detection. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 8495–8504
54. Snell J, Swersky K, Zemel RS (2017) Prototypical networks for few-shot learning. [arXiv:1703.05175](https://arxiv.org/abs/1703.05175)
55. Sutton RS, McAllester D, Singh S, et al (1999) Policy gradient methods for reinforcement learning with function approximation. *Adv Neural Inf Process Syst* 12
56. Villa-Pérez ME, Álvarez-Carmona MÁ, Loyola-González O et al (2021) Semi-supervised anomaly detection algorithms: a comparative summary and future research directions. *Knowl Based Syst* 218:106878
57. Vinyals O, Blundell C, Lillicrap T et al (2016) Matching networks for one shot learning. *Adv Neural Inf Process Syst* 29:3630–3638
58. Wang Y, Yao Q, Kwok JT et al (2020) Generalizing from a few examples: a survey on few-shot learning. *ACM Comput Surv (CSUR)* 53(3):1–34
59. Wyatt J, Leach A, Schmon SM, et al (2022) Anoddpm: anomaly detection with denoising diffusion probabilistic models using simplex noise. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 650–656
60. Xu H, Pang G, Wang Y et al (2023) Deep isolation forest for anomaly detection. *IEEE Trans Knowl Data Eng* 35(12):12591–12604
61. Yang L, Zhang Z, Song Y et al (2023) Diffusion models: a comprehensive survey of methods and applications. *ACM Comput Surv* 56(4):1–39

62. Yuan S, Zheng P, Wu X, et al (2020) Few-shot insider threat detection. In: Proceedings of the 29th ACM international conference on information and knowledge management, CIKM '20
63. Zamanzadeh Darban Z, Webb GI, Pan S et al (2024) Deep learning for time series anomaly detection: a survey. *ACM Comput Surv* 57(1):1–42
64. Zha D, Lai KH, Wan M, et al (2020) Meta-AAD: active anomaly detection with deep reinforcement learning. In: 2020 IEEE international conference on data mining (ICDM), pp 771–780
65. Zhang X, Xu Y, Lin Q, et al (2019) Robust log-based anomaly detection on unstable log data. In: Proceedings of the 2019 27th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering, pp 807–817
66. Zhong C, Gursoy MC, Velipasalar S (2019) Deep actor-critic reinforcement learning for anomaly detection. In: 2019 IEEE global communications conference (GLOBECOM)
67. Zhou C, Paffenroth RC (2017) Anomaly detection with robust deep autoencoders. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, pp 665–674
68. Zhou X, Liang W, Shimizu S et al (2021) Siamese neural network based few-shot learning for anomaly detection in industrial cyber-physical systems. *IEEE Trans Industr Inf* 17(8):5790–5798

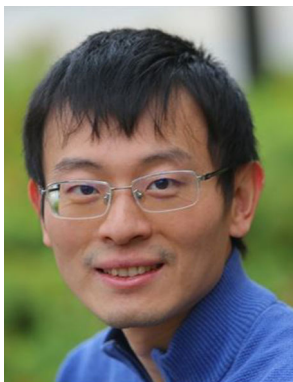
Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Xiao Han earned his Ph.D. from the Department of Computer Science at Utah State University. His research areas include data mining, machine learning, and artificial intelligence, with a particular focus on anomaly detection.



Depeng Xu is an assistant professor in the Department of Software & Information Systems and the School of Data Science at the University of North Carolina at Charlotte. His research focuses on data mining and machine learning, specifically differential privacy, algorithmic fairness, and ethical AI.



Shuhan Yuan is an assistant professor in the School of Computing at Utah State University. His research focuses on data mining and machine learning, specifically responsible anomaly detection.



Xintao Wu is a professor and the Charles D. Morgan/Axiom Endowed Graduate Research Chair in Database and leads Social Awareness and Intelligent Learning (SAIL) Lab in the Electrical Engineering and Computer Science Department at the University of Arkansas. His major research interests include data mining, data privacy, fraud detection, fairness-aware learning, and big data analysis.