Deep Anomaly Detection via Active Anomaly Search

Chao Chen*
National Key Laboratory for Novel Software Technology
School of Artificial Intelligence
Nanjing University
Nanjing, China
chenc@lamda.nju.edu.cn

Dawei Wang*
Alibaba Group
Hang Zhou, China
david.wdw@alibaba-inc.com

Feng Mao
Alibaba Group
Hang Zhou, China
maofeng.mf@alibaba-inc.com

Jiacheng Xu
National Key Laboratory for Novel Software Technology
School of Artificial Intelligence
Nanjing University
Nanjing, China
xujc@lamda.nju.edu.cn

Zongzhang Zhang†
National Key Laboratory for Novel Software Technology
School of Artificial Intelligence
Nanjing University
Nanjing, China
zzzhang@nju.edu.cn

Yang Yu
National Key Laboratory for Novel Software Technology
School of Artificial Intelligence
Nanjing University
Nanjing, China
yuy@nju.edu.cn

ABSTRACT

Anomaly detection (AD) holds substantial practical value, and considering the limited labeled data, the semi-supervised anomaly detection technique has garnered increasing attention. We find that previous methods suffer from insufficient exploitation of labeled data and under-exploration of unlabeled data. To tackle the above problem, we aim to search for possible anomalies in unlabeled data and use the searched anomalies to enhance performance. We innovatively model this search process as a Markov decision process and utilize a reinforcement learning algorithm to solve it. Our method, Deep Anomaly Detection and Search (DADS), integrates the exploration of unlabeled data and the exploitation of labeled data into one framework. Experimentally, we compare DADS with several state-of-the-art methods in widely used benchmarks, and the results show that DADS can efficiently search anomalies from unlabeled data and learn from them, thus achieving good performance.
Code: https://github.com/LAMDA-RL/DADS

KEYWORDS

Reinforcement Learning; Anomaly Detection; Deep Learning

ACM Reference Format:
Chao Chen*, Dawei Wang*, Feng Mao, Jiacheng Xu, Zongzhang Zhang†, and Yang Yu. 2024. Deep Anomaly Detection via Active Anomaly Search. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 9 pages.

*Equal contribution.
†Corresponding author.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

Anomaly detection (AD) [4] is a classical data mining task which aims at detecting data instances that significantly deviate from the majority. Nowadays, with the rapid development of information technology, there is an increasing demand for identifying abnormal data, and as such AD is playing a more and more important role in domains like cybersecurity, healthcare, and finance [7, 18, 29]. In this work, we focus on a specific type of AD, namely tabular semi-supervised AD, where the training dataset is composed of a small labeled dataset and a large unlabeled dataset. We think this kind of task is of great importance in real application scenarios since labeled data is more difficult to obtain than unlabeled data due to privacy, security, and other problems [30].

In classical semi-supervised AD tasks, both training and testing sets share the same distribution, which means labeled anomalies in the training set contain all classes of anomalies in the testing set. However, in real-world scenarios, it is common for incoming anomalies to originate from previously unseen distributions, necessitating the AD method to have enhanced capability in mining the unlabeled dataset to discover unknown anomaly classes for making accurate judgments.

We highlight several challenges that need to be addressed in the tabular semi-supervised AD domain:

- **Robust to contaminated unlabeled data**: Datasets from the real world often contain contaminated unlabeled data, however, state-of-the-art semi-supervised AD methods suffer robustness issues with large contamination ratio [23, 27].
- **Utilize known anomalies to detect unknown anomalies**: Anomalies from real-word datasets are sometimes diversified [20] and incoming new data may include unknown anomalies [26]. Semi-supervised AD methods need to leverage known anomalies to explore contaminated unlabeled data efficiently to find potential unknown anomalies.
- **Exploration-exploitation dilemma**: Anomalies are very rare in real-world datasets. On top of exploring unlabeled data, semi-supervised AD methods also need to exploit labeled known anomalies effectively.

To Address the above challenges, especially the problem of exploration-exploitation dilemma, Reinforcement Learning is a good alternative. Reinforcement learning (RL) [13] is a machine learning paradigm in which an agent learns an optimal policy through interacting with the environment. Since its birth, RL has attracted lots of attention and shown its effectiveness in many tasks. Deep RL (DRL) [1] takes a step further, which combines RL with deep learning [14] and achieves prominent performance in many fields such as games and automatic driving [17, 24, 25].

However, there is still little research on the application of RL to AD. A recently proposed RL-based AD method called DPLAN [20] uses an RL setting to explore unlabeled samples based on labeled anomalies. Experiments of DPLAN show that it is a successful trial of applying RL to the AD domain. However, through reproduction we find that DPLAN faces problems like overfitting and weak robustness to the contamination of unlabeled datasets.

Regardless of the disadvantages of DPLAN, we still believe that RL will become a powerful tool for AD with a carefully designed environment. In view of weak robustness to contamination, we propose that we can search for possible anomalies from unlabeled data with the help of labeled anomalies. The search mechanism can not only clean unlabeled data (distinguish anomalies from normal data) but also use searched anomalies to improve performance, thus making contamination no longer a problem. To implement the aforementioned search mechanism, we propose a more suitable Markov Decision Process (MDP) formalization. By modeling it as a sequential decision-making problem, the agent can learn to search for anomalies and then utilize them in the subsequent training process, which is unattainable through by the one-shot decision-making of bandit-like problem.

The main contributions of this work can be summarized as improvements in scenarios where the testing set contains unknown anomaly classes. With the help of RL, DADS integrates the search and refinement of unlabeled datasets into one model. Thus, the issues of contamination and unknown anomaly search existing in most AD methods are well solved, and our method achieves significantly better performance compared with several frequently-used semi-supervised AD methods.

2 BACKGROUND AND RELATED WORKS

In this section, we first introduce the basic of reinforcement learning (RL) and anomaly detection (AD), then present some classical or latest semi-supervised tabular AD methods.

2.1 Reinforcement Learning

In RL, the agent is asked to interact with the environment to deal with a sequential decision-making problem. At each time step t, the agent senses the state s_t from the environment and then selects an action a_t according to policy $\pi(a|s)$, given the state $s = s_t$. The action is passed to the environment and executed, which causes a transition to a new state s_{t+1} and a scalar reward r_t. The interaction continues until the task terminates. The objective of RL is to train such an agent to maximize the expected discounted cumulative reward $\mathbb{E}_\pi[\sum_t \gamma^t r_t]$, where $\gamma \in (0, 1]$ is the discount factor.

The standard RL can be formalized as a Markov decision process (MDP). It can be defined as a 4-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P} \rangle$, where $\mathcal{S}$ denotes state space; $\mathcal{A}$ denotes action space; $\mathcal{R}$ denotes reward function $\mathcal{S} \times \mathcal{A} \to \mathbb{R}, \mathcal{R}(s, a) = \mathbb{E}[r_t | s_t = s, a_t = a]$; and $\mathcal{P}$ denotes transition function $\mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1], \mathcal{P}(s, a, s') = p(s_{t+1} = s' | s_t = s, a_t = a)$. The whole process can be expressed as a trajectory: $\tau = (s_0, a_0, s_1, a_1 ... s_T)$. The optimization decision problem can be formalized as:

$$\max_\pi J(\pi) = \mathbb{E}_{\tau \sim p_\pi(\tau)}[r(\tau)], \tag{1}$$

where $r(\tau) = \sum_t \gamma^t r_t$ is the sum of discounted reward, $p_\pi(\tau) = p(s_0) \prod_{t=1}^{T} \pi(a_{t-1}|s_{t-1}) p(s_t|s_{t-1}, a_{t-1})$ is the distribution of trajectories.

Traditional RL algorithms can be divided into two categories, model-based [11, 16] and model-free [3]. In this paper we use a model-free RL algorithm SAC [10] as the baseline method. SAC adds an extra entropy term to RL's original target, and thus its objective function becomes:

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s_t, a_t) \sim \tau_\pi} \gamma^t \Big[r(s_t, a_t) + \alpha \mathcal{H}\big(\pi(\cdot \mid s_t)\big) \Big], \tag{2}$$

where π is represented as a policy network, (s_t, a_t) is a state-action pair within trajectory τ_π generated by policy π, $\mathcal{H}(\pi(. \mid s_t))$ is the entropy term which evaluates the uncertainty of policy π, and $\alpha > 0$ is a coefficient, by default $\alpha = 1$.

2.2 Semi-supervised Tabular Anomaly Detection

In this paper, we focus on semi-supervised tabular AD problems, where both unlabeled samples and a small portion of the labeled anomalies and labeled normal samples are available during training [22]. Generally, semi-supervised AD methods can be classified into supervised-based and unsupervised-based categories [9].

On top of the supervised learning paradigm, supervised-based semi-supervised AD methods add a bias term to describe the data characterization based on some assumptions. DevNet [19] optimizes the deviation loss, which requires the anomaly scores of normal data to follow a prior distribution, while the anomaly scores of anomalies to be far away from that distribution. The mentioned DPLAN is also supervised-based, since its reward function implies the cluster assumption [6]. A drawback of this kind of methods is that their performance relies heavily on the consistency between the distribution of datasets and the assumptions.

On the other hand, unsupervised-based semi-supervised AD methods extend the unsupervised AD methods by exploiting labeled data as a regularization term in the loss function. SSAD [9] and DeepSAD [23] are representatives of this category, both of them tries to map the data into a hypersphere while treating anomalies as an additional regularization term, requiring them to be distant from the sphere's center. DIAD [5] extends the explainable GA^{2}M model to the AD method with the Partial IDentification (PID) objective [8], and a differentiable AUC loss is added to incorporate labeled data. A recent work[12] proposes overlap loss, which aims at minimizing the overlap between the distribution of unlabeled data and the distribution of the anomalies. In essence, this is akin to using anomalies to regulate the distribution data. These methods also have problems like weak robustness to data contamination, insufficient exploration of unlabeled data, etc.

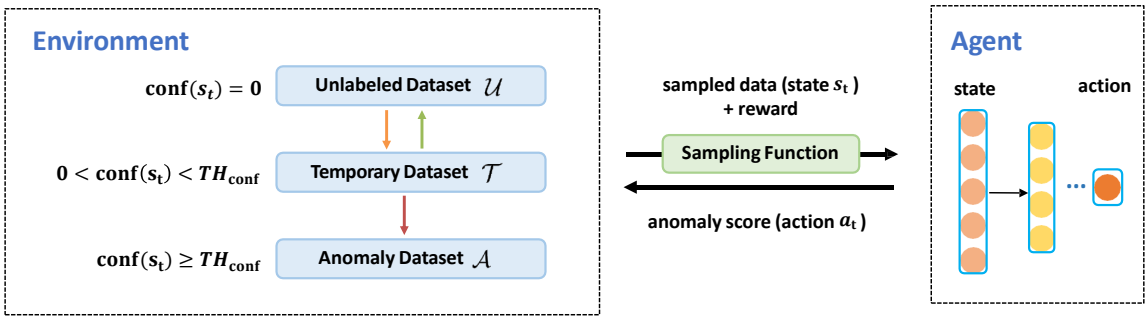


Figure 1: An illustration of our method DADS. See text for details.

3 OUR METHOD

Before introducing our method, we would like to give a formal statement of the task we aim at. Consider a semi-supervised AD scenario where two datasets are available:

- **Anomaly dataset** $\mathcal{D}^a$: a small dataset containing anomalies.
- **Unlabeled dataset** $\mathcal{D}^u$: a relatively large dataset containing both abnormal and normal data.

Given the above two datasets for training, we aim to find an anomaly scoring function $\Phi(\cdot) : \mathcal{D} \mapsto \mathbb{R}$, such that $\Phi(s_i) > \Phi(s_j)$, where s_i is abnormal and s_j is normal. In this paper, we simply focus on tabular data.

3.1 Algorithm Framework

Figure 1 is an illustration of DADS, which is mainly composed of two parts: an anomaly search environment and an anomaly scoring agent. The anomaly scoring agent is complemented based on the SAC algorithm. Given a piece of data, the agent will return the corresponding anomaly score, which is also the action in the MDP of the environment. Section 3.2 will give a detailed illustration of the anomaly scoring agent.

The anomaly search environment is a carefully designed MDP that continually searches for possible anomalies from unlabeled datasets. Below we define the state s and action a of the MDP of the environment. For transition function $\mathcal{P}$ and reward function $\mathcal{R}$, we will discuss them in Section 3.3 and Section 3.4 respectively.

- **State space** is the whole training dataset $\mathcal{D} = \{\mathcal{D}^a, \mathcal{D}^u\}$, with each $s_t \in \mathcal{D}$ sampled at timestep t be a state.
- **Action space** is a continuous space within range $[0, 1]$, with a_t corresponding to anomaly score of input data.

3.2 Anomaly Scoring Agent

As the name suggests, at each timestep t, the agent takes a piece of data s_t given by the environment, then its policy network π returns the action $a_t = \pi(s_t) \in [0, 1]$, which stands for the predicted anomaly score of s_t. Besides, we define that if $a_t > TH_{\text{score}}$, s_t is abnormal, otherwise normal, where $TH_{\text{score}} \in [0, 1]$ is a hyperparameter. By default, $TH_{\text{score}} = 0.7$

The agent is trained using the SAC algorithm. At inference time, the agent can directly return the anomaly score of any data s by calculating $\pi(s)$.

3.3 Sampling Function as State Transition

The sampling function of the environment is designed similar to that in previous work [20], which is composed of two parts: random sampling g_r and distance-based sampling g_d. The former aims at covering the whole training dataset, while the latter is to improve the search efficiency.

For random sampling, s_{t+1} is randomly selected from the whole training set $\mathcal{D}$, thus g_r can be expressed as $s_{t+1} \sim U(\mathcal{D})$, where U means equal distribution.

Distance-based sampling g_d can be expressed as:

$$g_d(s_{t+1} \mid s_t, a_t) = \begin{cases} \arg\min_{s \in \mathcal{D}'} d(s_t, s) & a_t > TH_{\text{score}} \\ \arg\max_{s \in \mathcal{D}'} d(s_t, s) & a_t \le TH_{\text{score}} \end{cases}, \quad (3)$$

where $\mathcal{D}' \subset \mathcal{D}$ is a randomly sampled subset, $d(s_t, s)$ calculates the Euclidean distance. We assume that the distance within abnormal data or normal data is smaller than that between them. If the agent judges the current data as an anomaly, then the environment will sample a piece of data close to it, otherwise far away from it. Thus, both two cases help explore possible anomalies in a sequential way.

During training, g_r and g_d are used with probabilities p and $1-p$. This way enables the agent to gain access to the full dataset while achieving higher search efficiency. p is set to 0.7 by default.

3.4 Reward Function for Anomaly Search

Before talking about the reward function, we would like to first introduce the confidence score used in our DADS, which is designed to make the environment more suitable for anomaly search. Initially, each data s in $\mathcal{D}$ is given an attribute $\texttt{conf}(s)$, representing the confidence of classifying s as an anomaly. For data $s \in \mathcal{D}^a$, $\texttt{conf}(s)$ is initialized to TH_{conf}; while for $s \in \mathcal{D}^u$, $\texttt{conf}(s)$ is initialized to 0. Here TH_{conf} is a hyperparameter greater than 0, which regulates the intensity of anomaly search.

In order to demonstrate the effect of conf more intuitively, as Figure 1 shows, the training dataset is partitioned into three datasets according to conf, respectively $\mathcal{A}$, $\mathcal{T}$, and $\mathcal{U}$. Anomaly dataset $\mathcal{A}$ contains data s satisfying $\texttt{conf}(s) \ge TH_{\text{conf}}$, which are searched or labeled anomalies; temporary dataset $\mathcal{T}$ contains data s satisfying $0 < \texttt{conf}(s) < TH_{\text{conf}}$, which needs further judgement; the remaining part of $\mathcal{D}$ is the unlabeled dataset. The confidence score keeps changing across the training according to the anomaly score given by the agent. There are two situations.

- $s_t \in \{\mathcal{U}, \mathcal{T}\}$ ($\texttt{conf}(s_t) < TH_{\text{conf}}$): if $a_t \geq TH_{\text{score}}$, $\texttt{conf}(s_t)$ will be added by 1 since it is currently judged as an anomaly, then based on updated $\texttt{conf}(s_t)$, s_t will be placed into $\mathcal{T}$ or $\mathcal{A}$ (orange and red arrows in Figure 1). If $a_t < TH_{\text{score}}$, $\texttt{conf}(s_t)$ will be directly set to 0 (green arrow in Figure 1).
- $s_t \in \mathcal{A}$ ($\texttt{conf}(s_t) \geq TH_{\text{conf}}$): here s_t is either a labeled anomaly or a searched anomaly, so we will not make any adjustments to its confidence. The agent is asked to correctly classify s_t as an anomaly, which will be discussed in the reward function in Section 3.4.

To sum up, given a_t, the update of $\texttt{conf}(s_t)$ can be written as:

$$\texttt{conf}(s_t) = \begin{cases} \texttt{conf}(s_t) + 1 & s_t \in \{\mathcal{U}, \mathcal{T}\}, a_t \geq TH_{\text{score}} \\ 0 & s_t \in \{\mathcal{U}, \mathcal{T}\}, a_t \leq TH_{\text{score}} \\ \texttt{conf}(s_t) & \text{else} \end{cases}. \quad (4)$$

The introduction of confidence score asks the environment to search for anomalies in a conservative way. Any data in unlabeled dataset $\mathcal{D}^a$ needs to be judged as abnormal TH_{conf} consecutive times before it can be added to the anomaly dataset. Notice that the agent is continuously updated, the TH_{conf} times judgment is actually an integration of the results of different agents, which resembles the idea of ensemble learning.

With the confidence score and the $\{\mathcal{A}, \mathcal{T}, \mathcal{U}\}$ partition of the training dataset, we can now give the reward function of the anomaly search environment, which is designed based on both external reward and intrinsic reward.

For each data sampled s_t, if it belongs to $\mathcal{A}$, the agent is asked to make a correct judgment. If the agent correctly classifies s_t as an anomaly, it will get a positive reward, otherwise, it will be punished. This is what we call external reward, and the effect of it can be regarded as simply fitting labeled anomalies.

However, simply using external reward is easy to cause overfitting due to the unilateral supervisory signal of labeled anomalies, especially when the labeled anomalies are limited. To better utilize the unlabeled data, we introduce the intrinsic reward. First, if s_t comes from $\mathcal{T}$ or $\mathcal{U}$, and is sampled randomly by g_r, it is likely that s_t is normal, so the agent will receive a small positive reward if it judges s_t as normal. Second, if the confidence of an unlabeled data s_t reaches TH_{conf} and s_t is put into $\mathcal{A}$, to encourage the search of possible anomalies, agent will receive a reward in proportion to the unsupervised anomaly score of s_t, which is denoted as $\texttt{UNSUP}(s_t) \in [0, 1]$. Here we choose Isolation Forest[15] as the unsupervised AD method. For the remaining conditions, the reward is 0.

To conclude, the reward function can be written as:

$$r(s_t|a_t) = \begin{cases} \alpha_1 \texttt{sgn}(a_t - TH_{\text{score}}) & \texttt{conf}(s_t) \geq TH_{\text{conf}} \\ \alpha_2 \texttt{UNSUP}(s_t) & \texttt{conf}(s_t) = TH_{\text{conf}} - 1, \\ & a_t \geq TH_{\text{score}}, \\ \alpha_3 & s_t \in \{\mathcal{U}, \mathcal{T}\}, s_t \sim U(\mathcal{D}), \\ & a_t < TH_{\text{score}} \\ 0 & \text{else} \end{cases}, \quad (5)$$

where $\alpha_1, \alpha_2, \alpha_3$ are hyperparameters for adjusting reward allocation, $\texttt{sgn}(x)$ is the modified indicator function, it returns 1 when $x > 0$, -1 otherwise. By default, $\alpha_1 = 1, \alpha_2 = 5, \alpha_3 = 0.5$.

3.5 Algorithm Overview

In the training phase (see Algorithm 1), we first initialize the actor network (π) and critic networks (q and v) of the SAC algorithm, together with an experience replay buffer (see lines 1~3). The whole training procedure is composed of n_episodes (see lines 4~18). In each episode, the agent does n_steps interaction with the environment. In each step of interaction (see lines 6~10), the agent gives anomaly score a_t of current sampled data s_t, then the environment makes adjustments and returns reward according to the s_t, and finally the environment samples next data s_{t+1}. The information of this step is then stored in replay buffer. In agent training (see lines 11~16), the agent first collects warmup_steps data with no parameter updating, after warmup_steps, the agent will update the parameters of the actor and critic networks according to the SAC algorithm. Finally, the trained actor network is returned.

In the inference phase, for each test data s, we can directly use the output of the agent $\pi(s; \theta)$ as its anomaly score.

Algorithm 1 Training of DADS

1: **Input:** $\mathcal{D} = \{\mathcal{D}^a, \mathcal{D}^u\}$
2: **Initialize:** $\pi(s; \theta)$, $q(s, a; \phi_1)$, $q(s, a; \phi_2)$, $v(s; \psi)$, $v(s; \overline{\psi})$
3: Experience replay buffer $\mathcal{D}$
4: **for** $i = 1$ to n_episodes **do**
5: **for** $t = 1$ to n_steps **do**
6: Agent takes action a_t according to $\pi(s_t; \theta)$
7: Environment adjusts confidence of s_t with Equation 4
8: Environment returns r_t according to Equation 5
9: Environment samples data s_{t+1} according to Section 3.3
10: Store (s_t, a_t, r_t, s_{t+1}) in the replay buffer $\mathcal{D}$
11: **if** total_steps > warmup_steps **then**
12: **for** $j = 1$ to update_times **do**
13: Sample a batch of transitions $\mathcal{B}$ from $\mathcal{D}$
14: Update $\theta, \phi_1, \phi_2, \psi, \overline{\psi}$ according to SAC algorithm
15: **end for**
16: **end if**
17: **end for**
18: **end for**
19: **Output:** $\pi(s; \theta)$

4 EXPERIMENTS

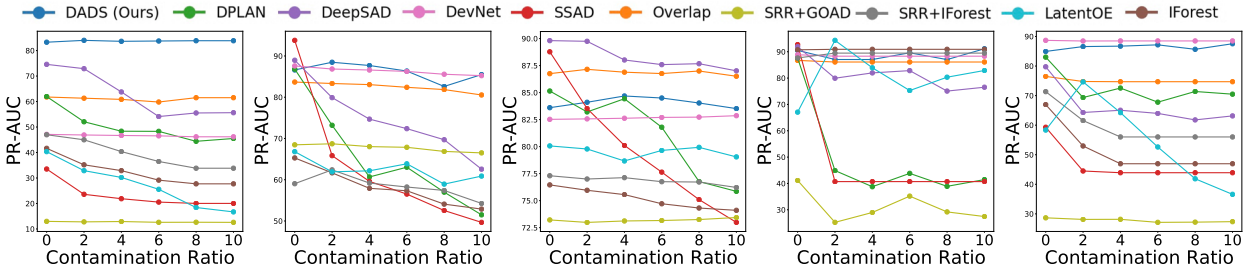
This section first gives a brief introduction to the datasets, evaluation metrics, and baseline methods, then gives experiment results on two scenarios. Comprehensive ablation study is also conducted to demonstrate the effect of different components. For details in the experiment, please refer to the Appendix (https://www.lamda.nju.edu.cn/chenc/DADS-Appendix.pdf).

4.1 Datasets and Evaluation Metrics

To make a comprehensive comparison between DADS and other methods, we choose 5 datasets with single anomaly class and 4 datasets with multiple anomaly classes. The former 5 of datasets are used in experiments of scenario 1, with dimension ranging from 6 to 36 and anomaly percentage ranging from 1.2% to 31.6%. To make further comparison, scenario 2 is designed using 4 datasets

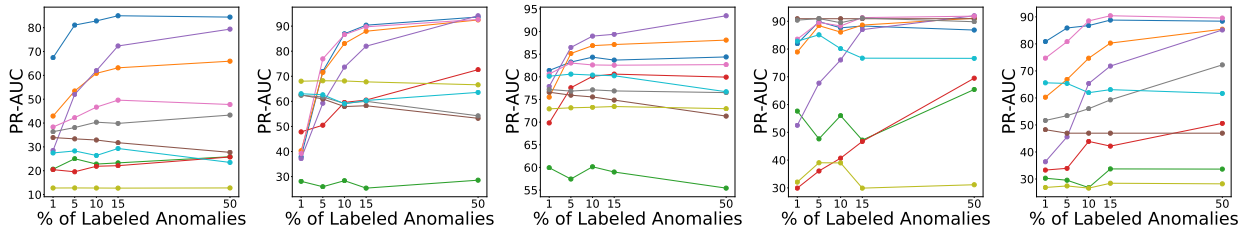
Table 1: AUC-PR performance (mean±std) of settings 1.1 and 2.1.

	DADS	DPLAN	DeepSAD	DevNet	SSAD	Overlap	SRR+GOAD	SRR+IForest	LatentOE	IForest
Known AD Scenario										
annthyroid	**0.839 ±0.026**	0.455 ±0.066	0.556 ±0.061	0.462 ±0.041	0.200 ±0.045	0.615 ±0.071	0.126 ±0.019	0.338 ±0.041	0.167 ±0.037	0.277 ±0.045
cardio	**0.855 ±0.079**	0.515 ±0.126	0.626 ±0.079	0.852 ±0.076	0.497 ±0.138	0.805 ±0.074	0.665 ±0.069	0.543 ±0.089	0.609 ±0.134	0.529 ±0.075
satellite	0.835 ±0.010	0.759 ±0.061	**0.87 ±0.036**	0.829 ±0.014	0.730 ±0.015	0.865 ±0.019	0.734 ±0.014	0.762 ±0.013	0.790 ±0.016	0.741 ±0.017
satimage2	**0.911 ±0.039**	0.415 ±0.205	0.765 ±0.153	0.883 ±0.086	0.407 ±0.112	0.861 ±0.110	0.274 ±0.096	0.895 ±0.033	0.829 ±0.121	0.909 ±0.031
thyroid	0.875 ±0.058	0.705 ±0.136	0.631 ±0.107	**0.885 ±0.066**	0.439 ±0.146	0.747 ±0.136	0.274 ±0.077	0.560 ±0.119	0.366 ±0.152	0.470 ±0.116
Average	**0.863 ±0.042**	0.570 ±0.119	0.690 ±0.087	0.782 ±0.057	0.455 ±0.091	0.779 ±0.082	0.415 ±0.055	0.620 ±0.059	0.552 ±0.092	0.585 ±0.057
Average_rank	**1.6**	6.6	4.2	3.0	9.0	3.0	8.6	5.6	7.0	6.4
Unknown AD Scenario										
multi_shuttle	**0.992 ±0.005**	0.595 ±0.144	0.912 ±0.049	0.943 ±0.012	oom	0.805 ±0.075	0.672 ±0.089	0.804 ±0.022	0.803 ±0.217	0.758 ±0.019
multi_cardio	**0.871 ±0.051**	0.544 ±0.058	0.812 ±0.043	0.807 ±0.050	0.704 ±0.04	0.826 ±0.068	0.605 ±0.044	0.534 ±0.051	0.751 ±0.109	0.593 ±0.024
multi_har	0.917 ±0.043	0.513 ±0.120	0.775 ±0.109	**0.918 ±0.016**	0.766 ±0.022	0.837 ±0.028	0.554 ±0.127	0.718 ±0.031	0.625 ±0.202	0.722 ±0.027
multi_annthyroid	**0.729 ±0.104**	0.298 ±0.076	0.216 ±0.091	0.239 ±0.091	0.142 ±0.042	0.300 ±0.133	0.092 ±0.016	0.144 ±0.030	0.127 ±0.018	0.133 ±0.024
Average	**0.877 ±0.051**	0.488 ±0.100	0.679 ±0.073	0.727 ±0.043	0.537 ±0.035	0.692 ±0.076	0.481 ±0.069	0.550 ±0.033	0.577 ±0.136	0.552 ±0.024
Average_rank	**1.25**	7.75	3.75	2.75	6.0	2.75	8.5	7.0	7.0	7.25



(a) annthyroid **(b) cardio** **(c) satellite** **(d) satimage2** **(e) thyroid**

Figure 2: AUC-PR of DADS and baselines in setting 1.1.



(a) annthyroid **(b) cardio** **(c) satellite** **(d) satimage2** **(e) thyroid**

Figure 3: AUC-PR of DADS and baselines in setting 1.2.

with multiple anomaly classes, with size ranging from thousands to tens of thousands and dimension ranging from 9 to 561. For detailed information of the datasets, please refer to Appendix E.

The mentioned datasets serve as base pool of our experiments. We set two adjustable parameters, namely anomalies_ratio and contamination_ratio. anomalies_ratio corresponds to the ratio of known anomalies to total anomalies (in scenario 2 total anomalies means anomalies of the known anomaly class); contamination_ratio corresponds to the percentage of anomalies in unlabeled data.

Each dataset is splited into training set, validation set, and testing set, each accounting for 60%, 20%, and 20% of the total dataset with 10 different random seeds. Validation set and testing set are fixed after splitting, so the percentage of each class is consistent with original dataset. Training dataset is manually generated according to two adjustable parameters: anomalies_ratio and contamination_ratio, and, as stated before, it is composed of $\mathcal{D}^a$ and $\mathcal{D}^u$.

We select Area Under the Precision-Recall Curve (AUC-PR) and Area Under the Receiver Operating Characteristic Curve (AUC-ROC) as the evaluation metric. The reported results are averaged over 10 random seeds. Due to space limitation, below we only show the AUC-PR results, for AUC-ROC, please refer to Appendix F.

4.2 Competing Methods

SSAD [9] and Deep SAD [23] are the early works in this field, both of them takes the idea o using hypersphere to envelop the normal data and exclude the anomalies. DevNet [19] introduces deviation loss and combines it with supervised loss, which is a simple yet effective semi-supervised AD method. Overlap [12] is a recent method that proposes a novel overlap loss, which shows effect across various datasets. We reproduce DPLAN [20] and include it in our baselines. Considering that the distance calculation step in DPLAN is time-consuming, sampling size is decreased from

Table 2: AUC-PR performance (mean±std) of settings 1.2 and 2.2.

	DADS	DPLAN	DeepSAD	DevNet	SSAD	Overlap	SRR+GOAD	SRR+IForest	LatentOE	IForest
Known AD Scenario										
annthyroid	**0.811 ±0.076**	0.250 ±0.076	0.520 ±0.067	0.422 ±0.062	0.196 ±0.047	0.534 ±0.092	0.128 ±0.016	0.381 ±0.043	0.283 ±0.052	0.334 ±0.050
cardio	0.719 ±0.147	0.260 ±0.130	0.592 ±0.136	**0.769 ±0.134**	0.505 ±0.149	0.715 ±0.138	0.682 ±0.064	0.617 ±0.073	0.626 ±0.056	0.610 ±0.070
satellite	0.833 ±0.024	0.574 ±0.092	**0.865 ±0.019**	0.831 ±0.015	0.776 ±0.014	0.851 ±0.020	0.732 ±0.014	0.769 ±0.013	0.806 ±0.013	0.759 ±0.018
satimage2	0.899 ±0.033	0.476 ±0.278	0.677 ±0.282	0.896 ±0.090	0.361 ±0.102	0.884 ±0.073	0.390 ±0.123	**0.910 ±0.037**	0.851 ±0.103	0.909 ±0.031
thyroid	**0.859 ±0.061**	0.295 ±0.158	0.456 ±0.111	0.808 ±0.141	0.340 ±0.092	0.668 ±0.231	0.274 ±0.080	0.534 ±0.120	0.654 ±0.126	0.470 ±0.116
Average	**0.824 ±0.068**	0.371 ±0.147	0.622 ±0.123	0.745 ±0.088	0.435 ±0.081	0.731 ±0.111	0.441 ±0.059	0.642 ±0.057	0.644 ±0.070	0.616 ±0.057
Average_rank	**2.0**	9.0	5.2	3.0	8.4	3.0	8.4	4.8	5.4	5.8
Unknown AD Scenario										
multi_shuttle	**0.991 ±0.004**	0.750 ±0.072	0.893 ±0.072	0.843 ±0.168	oom	0.884 ±0.094	0.715 ±0.059	0.852 ±0.016	0.937 ±0.046	0.819 ±0.014
multi_cardio	0.863 ±0.057	0.379 ±0.058	0.855 ±0.052	0.883 ±0.061	0.757 ±0.041	0.827 ±0.090	0.657 ±0.048	0.561 ±0.049	**0.889 ±0.092**	0.653 ±0.043
multi_har	0.934 ±0.047	0.562 ±0.138	0.830 ±0.107	0.948 ±0.016	**0.965 ±0.007**	0.879 ±0.031	0.619 ±0.060	0.741 ±0.034	0.556 ±0.119	0.756 ±0.033
multi_annthyroid	**0.676 ±0.148**	0.195 ±0.122	0.205 ±0.081	0.192 ±0.089	0.144 ±0.049	0.229 ±0.082	0.093 ±0.017	0.164 ±0.037	0.166 ±0.017	0.159 ±0.033
Average	**0.866 ±0.064**	0.472 ±0.097	0.696 ±0.078	0.717 ±0.084	0.622 ±0.033	0.705 ±0.074	0.521 ±0.046	0.579 ±0.034	0.637 ±0.068	0.597 ±0.031
Average_rank	**2.0**	7.75	3.75	3.75	5.33	3.75	8.5	7.0	4.75	7.25

1000 to 100. LatentOE[21], SRR [27], and Isolation Forest [15] are unsupervised AD methods. LatentOE is also a recent work targeting at solving contamination issue, it assumes anomalies in dataset and iteratively infers label of each data during training. SRR takes the idea of data refinement and improves the robustness of one-class classification in contaminated datasets. In our experiment SRR is combined with both Isolation Forest and the self-supervised representation learning method GOAD [2]. Isolation Forest is a classical unsupervised AD method, and is also used in the reward function of DADS. Details about the implementation of DADS and baselines are in Appendix B and C.

4.3 Scenario 1: Experiments on Datasets with Single Anomaly Class

In this scenario, we use 5 datasets with single anomaly class to compare DADS with baseline methods. As stated above, there are two adjustable parameters: anomalies_ratio and contamination_ratio, we will fix one at a time and adjust another. The results of these two groups of settings are as follows.

Setting 1.1: AUC versus incremental contamination ratio with fixed anomalies ratio. First we set anomalies_ratio to 0.1, which means the known anomalies account for 10% of total anomalies. Next, we gradually increase contamination_ratio within the range $[0, 0.1]$ to test whether DADS and baselines are robust to pollution in unlabeled data. Here we report the result of DADS and baseline methods when contamination_ratio = 0.1 in the upper half of Table 1, and the average rank of each method averages its ranking of results on different datasets. DADS ranks first in both average AUC-PR and average rank, which shows the advantage of DADS in resisting high contamination in unlabeled data. From Figure 2 we can also see that as the contamination ratio goes up, the performance of DADS hardly declines, which again shows the robustness of DADS.

Setting 1.2: AUC versus incremental anomalies ratio with fixed contamination ratio. In this part, we set contamination_ratio to 0.04 and adjust anomalies_ratio within the range $[0.01, 0.5]$ to see whether the search of possible anomalies makes contribution to the performance of DADS. Table 2 shows the results when anomalies_ratio = 0.05, and the upper half corresponds to this setting, graphic results are in Figure 3. From Table 2 we can see that DADS performs well when only a little labeled anomalies are available. In addition, Figure 3 shows that the performance of DADS increases steadily with the increase of anomalies_ratio.

4.4 Scenario 2: Experiments on Datasets with Multiple Anomaly Classes

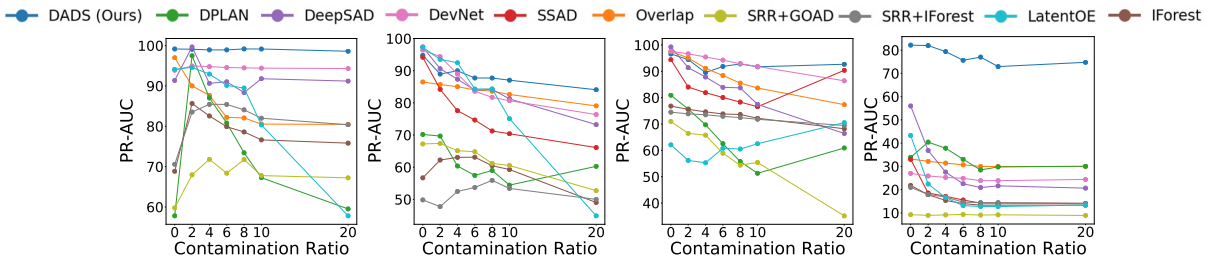
The above experiments on datasets with single anomaly class show the advantage of DADS in both the exploration of unlabeled dataset and robustness to contaminated data. When testing on datasets with multiple anomaly classes, we set one anomaly as the known anomaly class while leaving others as unknown. We believe that the tasks in this scenario will be more difficult and can better show the advantages of DADS.

Setting 2.1: AUC versus incremental contamination ratio with fixed anomalies ratio. Like setting 1.1, anomalies_ratio is set to 0.1, but the range of contamination_ratio is expanded from $[0, 0.1]$ to $[0, 0.2]$, which may test the robustness to contamination ratio of each method more extremely. An obvious phenomenon in Figure 4 is that as contamination_ratio goes up, all methods experience different degrees of decline, among which DADS shows the smallest performance reduction. This again shows the robustness of DADS to contamination. The lower half of Table 1 summarizes the results when contamination_ratio = 0.1. The performance advantage of DADS is further distinguished compared with scenario 1.

Setting 2.2: AUC versus incremental anomalies ratio with fixed contamination ratio. Here we fix contamination_ratio to 0.04 and increase anomalies_ratio from 0.01 to 0.5. Average AUC-PR results and rank when anomalies_ratio = 0.05 are shown in the lower half of Table 2. DADS still performs the best in average AUC-PR and average rank. We believe this should be attributed to the efficient search mechanism, which enables DADS to search for possible anomalies when there are multiple anomaly classes. Figure 5 shows graphic results, we can see that the performance of DADS increases steadily as the number of labeled anomalies goes up. In addition, DADS can achieve a high score even when only a few anomalies are known (e.g., 1%, 5%), which reflects the capability of DADS in utilizing both labeled anomalies and unlabeled data.

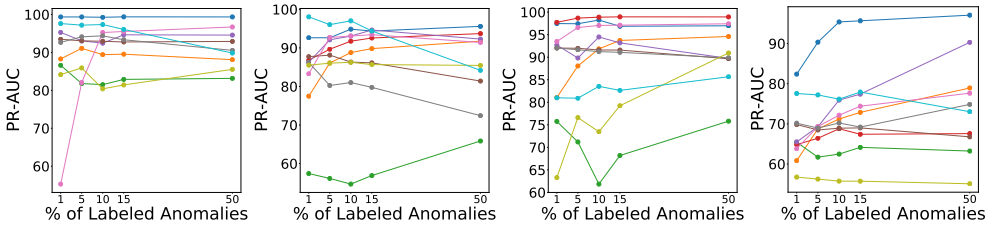
4.5 Analysis of Experiment Results

From the experiments conducted on the two scenarios, we can summarize the following empirical results.



(a) multi_shuttle (b) multi_cardio (c) multi_har (d) multi_annthyroid

Figure 4: AUC-PR of DADS and baselines in setting 2.1.



(a) multi_shuttle (b) multi_cardio (c) multi_har (d) multi_annthyroid

Figure 5: AUC-PR of DADS and baselines in setting 2.2.

4.5.1 DADS vs. DPLAN. We think it is necessary to make a comprehensive comparison between DADS and DPLAN, since they are all AD methods based on RL. According to the experiment results, DADS outperforms DPLAN in all two scenarios. To be specific, in settings 1.1 and 2.1, especially datasets like anthyroid, thyroid, and multi-har, the superiority of robustness to contamination of DADS is very obvious. In settings 1.2 and 2.2, DADS outperforms DPLAN when only small amount of labeled anomalies is known, which showcase the role of the search mechanism.

4.5.2 DADS vs. unsupervised AD methods. In terms of robustness, DADS has obvious advantage over methods like LatentOE and SRR+GOAD, while performs roughly the same compared with SRR+IForest and IForest. But the reasons for robustness are different. For unsupervised AD methods, they are largely label-independent, so the label change in dataset will not do much harm to performance. While for DADS, the robustness come from its search mechanism. In terms of AUC-PR performance, DADS outperforms unsupervised AD methods in all datasets except satimage2 and multi_cardio, this is very natural since DADS can utilize supervised signal from labeled anomalies while unsupervised AD methods cannot.

4.5.3 DADS vs. semi-supervised AD methods. Compared with semi-supervised methods like DeepSAD, SSAD, DevNet, and Overlap, DADS outperforms them in all four settings. Figures 2 and 3 indicate that DADS is more robust to data contamination, especially when compared with supervised-based methods like DPLAN. In settings 1.2 and 2.2, unsupervised-based methods like SSAD and Deep SAD need sufficient supervised signal to achieve relatively good performance (e.g., anomalies_ratio $\geq$ 0.15). While in contrast, DADS can achieve a high AUC-PR even when only little labeled anomalies are available (e.g., anomalies_ratio = 0.05), which may give credit to the possible anomalies found by the search mechanism.

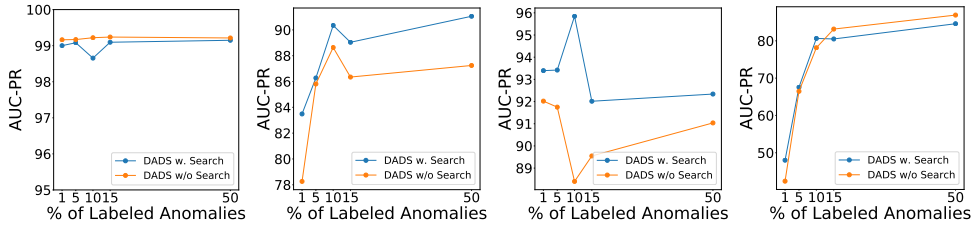
4.6 Ablation Study

Here we show the ablation study on three most important hyperparameters, TH_{conf}, p and α_2. Considering DADS's potential in dealing with datasets with multiple anomaly classes, ablation study is conducted under setting 2.2. For ablation study on other hyperparameters, please refer to Appendix G.

4.6.1 Ablation on TH_{conf}. To identify to what extent the search mechanism helps, we remove the search mechanism of DADS and leave other components unchanged by setting TH_{conf} to 100.

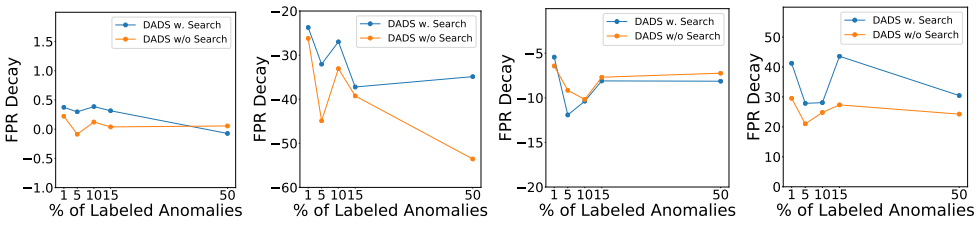
Figure 6 shows the AUC-PR of DADS and its variant, where DADS w. Search represents the original implementation of DADS, while DADS w/o Search stands for DADS without search. We can see that the search mechanism brings improvement in multi_cardio, multi_har and multi_annthyroid, while a slight performance decrease is observed in multi_shuttle. We think this may be because DADS w/o Search has already performed well in multi_shuttle, so that the improvement brought by the searched anomalies is limited, meanwhile any inaccuracies in the searched anomalies could hinder performance. For other datasets, there still exists substantial room for performance improvement, especially when the labeled anomalies are extremely limited.

To further demonstrate the effectiveness of search in detecting unknown anomalies, we introduce the FPR@TPR95_decay metrics, which is similar to the metrics used in previous work [28]. To be more specific, test dataset is divided into two datasets, the first containing normal data and anomalies of known classes, and the second containing normal data and anomalies of unknown classes. False Positive Rate at True Positive Rate 95%(FPR@TPR95) are calculated on these two datasets respectively and get FPR@TPR95_known, FPR@TPR95_unknown. Finally, FPR@TPR95_decay is calculated by FPR@TPR95_unknown - FPR@TPR95_known. By comparing



(a) multi_shuttle (b) multi_cardio (c) multi_har (d) multi_annthyroid

Figure 6: AUC-PR of DADS and its variant in setting 2.2.



(a) multi_shuttle (b) multi_cardio (c) multi_har (d) multi_annthyroid

Figure 7: FPR@TPR95_decay of DADS and its variant in setting 2.2.

the performance of DADS on test set with known anomalies and test set with unknown anomalies, FPR@TPR95_decay measures the performance drop caused by unknown anomalies. Considering the inverse correlation between FPR@TPR95 and performance, the higher FPR@TPR95_decay is, the better the capability is in searching unknown anomalies.

Figure 7 shows FPR@TPR95_decay of DADS and its variant. In multi_shuttle, multi_cardio, and multi_annthyorid, the search mechanism significantly reduces performance loss on unknown classes, while in multi_har, there's not much difference.

4.6.2 Ablation on p. Since p is an important hyperparameter for balancing random sampling and distance-based sampling, it's also worth testing how different value of p affects the performance of DADS. Same as Table 5, we set contamination_ratio = 0.04, anomalies_ratio = 0.05, and p is set from 0 to 1, the AUC-PR result is in table 3. It can be concluded that both random sampling and distance-based sampling are important, the absence of either one will have a significant impact on the performance.

4.6.3 Ablation on α_2. As an important hyperparameter in the reward function of DADS, $alpha_2$ regulates the reward the agent receives when finding a possible anomaly. We vary α_2 from 1 to 10 and record AUC-PR in Table 4, still, contamination_ratio = 0.04, anomalies_ratio = 0.05. We can see that an appropriate value of α_2 is important for unleashing the search mechanism of DADS. A too low value will result in a reduction in the intensity of search, while a too high value will lead to overly aggressive exploration.

5 CONCLUSION AND FUTURE WORK

In this paper, we present an RL-based semi-supervised tabular AD method DADS. By designing an anomaly search environment and a simple RL agent, our method can automatically balance between exploiting labeled dataset and exploring unlabeled dataset. DADS performs well across various settings, especially on datasets with multiple anomaly classes, which demonstrates its effectiveness.

For future work, though the inference stage is quick, DADS still consumes much time in the training phase due to the use of RL, which needs further improvement. In addition, there is still much room for improvement in the use of unsupervised signals. Lastly, future work could explore incorporating the search mechanism in the action space design. For example, action can not only contain the anomaly score of current data but also encompass the search direction for the next step.

Table 3: Ablation study on p (AUC-PR)

	p=0.0	p=0.3	p=0.5	p=0.7	p=1.0
multi_shuttle	0.971 ±0.014	0.986 ±0.007	0.986 ±0.008	**0.991 ±0.004**	**0.991 ±0.004**
multi_cardio	0.861 ±0.053	0.854 ±0.053	0.879 ±0.062	**0.863 ±0.057**	0.831 ±0.059
multi_har	0.904 ±0.052	0.907 ±0.051	0.905 ±0.051	**0.934 ±0.047**	0.906 ±0.068
multi_annthyroid	0.540 ±0.093	**0.688 ±0.107**	0.628 ±0.142	0.676 ±0.148	0.636 ±0.196
Average	0.819 ±0.053	0.859 ±0.055	0.850 ±0.066	**0.866 ±0.064**	0.841 ±0.082

Table 4: Ablation study of α_2 (AUC-PR)

	α_2 = 1.0	α_2 = 2.0	α_2 = 3.0	α_2 = 5.0	α_2 = 10.0
multi_shuttle	0.992 ±0.002	0.991 ±0.003	**0.993 ±0.003**	0.991 ±0.004	0.991 ±0.005
multi_cardio	0.862 ±0.061	0.861 ±0.050	**0.863 ±0.061**	**0.863 ±0.057**	0.836 ±0.086
multi_har	0.927 ±0.050	**0.934 ±0.050**	0.920 ±0.057	**0.934 ±0.047**	0.921 ±0.048
multi_annthyroid	0.614 ±0.147	0.631 ±0.161	0.662 ±0.097	**0.676 ±0.148**	0.630 ±0.142
Average	0.849 ±0.065	0.854 ±0.066	0.860 ±0.055	**0.866 ±0.064**	0.845 ±0.070

ACKNOWLEDGMENTS

We express our gratitude to the reviewers for their insightful and valuable comments. Special thanks to Zijing Wang and Weijian Liao for their helpful discussions and support. This work is supported by the NSFC (No. 62276126) and Alibaba Group through Alibaba Research Fellowship Program.

REFERENCES

[1] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. 2017. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine* 34, 6 (2017), 26–38.

[2] Liron Bergman and Yedid Hoshen. 2020. Classification-based anomaly detection for deneral data. In *International Conference on Learning Representations*.

[3] Sinan Çalışır and Meltem Kurt Pehlivanoğlu. 2019. Model-free reinforcement learning algorithms: A survey. In *2019 27th signal processing and communications applications conference (SIU)*. IEEE, 1–4.

[4] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *Comput. Surveys* 41, 3 (2009), 1–58.

[5] Chun-Hao Chang, Jinsung Yoon, Sercan Arik, Madeleine Udell, and Tomas Pfister. 2022. Data-efficient and interpretable tabular anomaly detection. *arXiv preprint arXiv:2203.02034* (2022).

[6] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. 2009. Semi-supervised learning. *IEEE Transactions on Neural Networks* 20, 3 (2009), 542–542.

[7] Arthur Charpentier, Romuald Elie, and Carl Remlinger. 2021. Reinforcement learning in economics and finance. *Computational Economics* (2021), 1–38.

[8] Parikshit Gopalan, Vatsal Sharan, and Udi Wieder. 2019. Pidforest: anomaly detection via partial identification. *Advances in Neural Information Processing Systems* 32 (2019).

[9] Nico Görnitz, Marius Kloft, Konrad Rieck, and Ulf Brefeld. 2013. Toward supervised anomaly detection. *Journal of Artificial Intelligence Research* 46 (2013), 235–262.

[10] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*. 1861–1870.

[11] Chengxing Jia, Fuxiang Zhang, Tian Xu, Jing-Cheng Pang, Zongzhang Zhang, and Yang Yu. 2024. Model gradient: unified model and policy learning in model-based reinforcement learning. *Frontiers of Computer Science* 18, 4 (2024), 184339.

[12] Minqi Jiang, Songqiao Han, and Hailiang Huang. 2023. Anomaly Detection with Score Distribution Discrimination. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (, Long Beach, CA, USA,) *(KDD '23)*. Association for Computing Machinery, New York, NY, USA, 984–996.

[13] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. 1996. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4 (1996), 237–285.

[14] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.

[15] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *IEEE International Conference on Data Mining*. 413–422.

[16] Fan-Ming Luo, Tian Xu, Hang Lai, Xiong-Hui Chen, Weinan Zhang, and Yang Yu. 2024. A survey on model-based reinforcement learning. *Science China Information Sciences* 67, 2 (2024), 121101.

[17] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[18] Thanh Thi Nguyen and Vijay Janapa Reddi. 2023. Deep reinforcement learning for cyber security. *IEEE Transactions on Neural Networks and Learning Systems* 34, 8 (2023), 3779–3795.

[19] Guansong Pang, Chunhua Shen, and Anton van den Hengel. 2019. Deep anomaly detection with deviation networks. In *ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 353–362.

[20] Guansong Pang, Anton van den Hengel, Chunhua Shen, and Longbing Cao. 2021. Toward deep supervised anomaly detection: Reinforcement learning from partially labeled anomaly data. In *ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1298–1308.

[21] Chen Qiu, Aodong Li, Marius Kloft, Maja Rudolph, and Stephan Mandt. 2022. Latent outlier exposure for anomaly detection with contaminated data. In *International Conference on Machine Learning*. PMLR, 18153–18167.

[22] Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. 2021. A unifying review of deep and shallow anomaly detection. *Proc. IEEE* (2021), 756–795.

[23] Lukas Ruff, Robert A. Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. 2019. Deep Semi-Supervised Anomaly Detection. *CoRR* abs/1906.02694 (2019).

[24] Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. 2017. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging* 2017, 19 (2017), 70–76.

[25] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy P.Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. 2017. Mastering the game of Go without human knowledge. *Nature* 550, 7676 (2017), 354–359.

[26] Miryam Elizabeth Villa-Pérez, Miguel Á Álvarez-Carmona, Octavio Loyola-González, Miguel Angel Medina-Pérez, Juan Carlos Velazco-Rossell, and Kim-Kwang Raymond Choo. 2021. Semi-supervised anomaly detection algorithms: A comparative summary and future research directions. *Knowledge-Based Systems* 218 (2021), 106878.

[27] Jinsung Yoon, Kihyuk Sohn, Chun-Liang Li, Sercan O Arik, Chen-Yu Lee, and Tomas Pfister. 2021. Self-supervise, refine, repeat: Improving unsupervised anomaly detection. *arXiv preprint arXiv:2106.06115* (2021).

[28] Jinsung Yoon, Kihyuk Sohn, Chun-Liang Li, Sercan O Arik, and Tomas Pfister. 2022. SPADE: Semi-supervised Anomaly Detection under Distribution Mismatch. *arXiv preprint arXiv:2212.00173* (2022).

[29] Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. 2021. Reinforcement learning in healthcare: A survey. *Comput. Surveys* 55, 1 (2021), 1–36.

[30] Shixiang Zhu, Henry Shaowu Yuchi, Minghe Zhang, and Yao Xie. 2019. Sequential adversarial anomaly detection for one-class event data. *arXiv preprint arXiv:1910.09161* (2019).