DRTA: Dynamic Reward Scaling for Reinforcement Learning in Time Series Anomaly Detection

1st Bahareh Golchin
School of Computer Science
Portland State University
Portland, Oregon, United States
bgolchin@pdx.edu

2nd Banafsheh Rekabdar
School of Computer Science
Portland State University
Portland, Oregon, United States
rekabdar@pdx.edu

3rd Kunpeng Liu
School of Computer Science
Portland State University
Portland, Oregon, United States
kunpeng@pdx.edu

***Abstract*— Anomaly detection in time series data is important for applications in finance, healthcare, sensor networks, and industrial monitoring. Traditional methods usually struggle with limited labeled data, high false-positive rates, and difficulty generalizing to novel anomaly types. To overcome these challenges, we propose a reinforcement learning-based framework that integrates dynamic reward shaping, Variational Autoencoder (VAE), and active learning, called DRTA. Our method uses an adaptive reward mechanism that balances exploration and exploitation by dynamically scaling the effect of VAE-based reconstruction error and classification rewards. This approach enables the agent to detect anomalies effectively in low-label systems while maintaining high precision and recall. Our experimental results on the Yahoo A1 and Yahoo A2 benchmark datasets demonstrate that the proposed method consistently outperforms state-of-the-art unsupervised and semi-supervised approaches. These findings show that our framework is a scalable and efficient solution for real-world anomaly detection tasks.**[1]

***Index Terms*—Time Series Anomaly Detection, Deep Reinforcement Learning, Variational Autoencoders, Active Learning, Dynamic Reward Scaling, Adaptive Rewards, Generative AI**

I. Introduction

Anomaly detection in time series is crucial across various domains, including data centers, sensor networks, cyber-physical systems, healthcare, demand forecast, and finance [27], [25], [8], and [31]. Most existing algorithms in the literature require manually tuning the problem parameters and selecting features. However, this is often tailored to specific data characteristics. Additionally, identifying anomalies manually is both time-consuming and labor-intensive, and it is prone to human error. In today's world, given the vast amounts of data involved, an automated system is essential for efficiently detecting anomalies in large-scale time series datasets [5].

In practice, anomaly detection faces two key challenges: first, anomalies are rare. Therefore, it is difficult to train models effectively. Second, real-world data is often time-dependent. To address the former challenge, many anomaly detection algorithms have been developed, which use unsupervised approaches that do not rely on labeled data. However, these methods are not generic. They often operate based on specific assumptions about anomaly patterns in the data, and these assumptions may not always hold in real-world scenarios, which would result in high false-positive rates. This issue stems from diverse user perspectives and varying definitions of anomalies [18], and [35].

To overcome this issue, supervised methods have also been developed to detect anomalies, and they perform well when ample labeled data is available. However, they struggle in situations with limited or no labels. Even with labeled data, these algorithms assume a stable underlying distribution. When the distribution is shifted for any reasons, they require retraining to maintain accuracy [34].

Moreover, semi-supervised learning algorithms have also been developed to detect anomalies in time series datasets. However, these methods are limited in scenarios where diverse anomaly types are present. The reason is that semi-supervised learning algorithms rely on a small set of known anomalies, and they may fail to identify novel anomalies in unlabeled data. Consequently, they are ineffective at detecting new types of anomalies [14].

To address the aforementioned challenges of weakly-supervised anomaly detection in time series data, in this paper, we propose using Deep Reinforcement Learning (DRL). A key aspect of Reinforcement Learning (RL) is the balance between exploration and exploitation [28]. Exploitation refers to selecting actions based on existing knowledge to maximize rewards. Exploration involves trying new actions to discover potentially better strategies. The agent must continuously navigate this trade-off to balance the immediate optimal decisions with the need to acquire more knowledge.

The DRL model in our study is designed to effectively use a limited set of labeled anomalous data (D_{la}) while extensively exploring a large pool of unlabeled data (D_u). This approach enables us to detect new anomaly types not present in the labeled dataset. The exploration process is improved by incorporating a Variational Autoencoder (VAE), which strengthens the DRL framework.

Additionally, given the high cost and limited availability of fully labeled data in real-world scenarios, our system uses active learning. This allows the RL agent to: 1) efficiently explore the environment and gather experience, and 2) make informed decisions based on the knowledge gained during exploration.

[1]Code is available at GitHub Repository.

2025 Conference on AI, Science, Engineering, and Technology (AIxSET)

Finally, the core of our DRL agent is a Long Short-Term Memory (LSTM) network, which 1) models sequential time series data and 2) captures long-term dependencies between events [10]. To improve the learning efficiency of the RL agent, we use the novel idea of reward shaping, which provides additional guidance to the reinforcement learning process. By designing a well-structured reward function, we help the RL agent learn meaningful patterns more quickly. This reduces the time needed to converge to an optimal policy. Reward shaping enables the agent to receive intermediate feedback. This is particularly helpful in complex environments where rewards are sparse or delayed. This approach prevents the agent from relying only on trial-and-error exploration. It instead encourages the agent to follow a more informed learning trajectory, which ultimately improves anomaly detection performance in time series data [22].

To summarize, our key contributions include the following.

- **Dynamic Reward Mechanism for Adaptive Exploration and Exploitation**. This study introduces a novel dynamic reward shaping mechanism that adaptively balances exploration and exploitation, enabling the RL agent to achieve robust anomaly detection performance.
- **Efficient Active Learning Integration with Adaptive Querying**. Our proposed method improves active learning by dynamically querying the most uncertain samples during training. This, combined with adaptive reward shaping, reduces the need for labeled data.
- **Unified Framework for Semi-Supervised Anomaly Detection**. By integrating RL with the dynamic reward function, active learning, and VAE, this study proposes a unified framework that effectively handles low-label systems.

This paper is structured as follows: Section II surveys relevant literature, Section III establishes the theoretical foundation for time series anomaly detection, Section IV describes our proposed methodology, Section V analyzes the implementation process, and Section VI presents our conclusions.

II. Related Work

In this section, we review machine learning-based approaches in the literature for time series anomaly detection.

A. Machine Learning-Based Anomaly Detection

Machine learning approaches address the limitations of statistical methods by using labeled training data to distinguish between normal and anomalous instances through classification or clustering techniques.

Supervised and Unsupervised Learning. Commonly used algorithms include Bayesian networks [3], support vector machines (SVM) [19], rule-based systems [30], and neural networks [23]. Clustering techniques, such as k-means, have also been used for anomaly detection [26].

Reinforcement Learning in Anomaly Detection. More recently, RL has been used in anomaly detection due to its ability to learn from interactions with the environment. Bourdonnaye et al. proposed using convolutional autoencoders within an RL framework to detect anomalies [2]. Huang et al. introduced a value-based DRL approach using the Deep Q-Network (DQN) algorithm [13]. Furthermore, reward shaping has emerged as an important technique in RL, particularly for enhancing anomaly detection systems. For instance, Devidze et al. introduce Exploration-Guided Reward Shaping, a self-supervised framework that accelerates reinforcement learning in sparse-reward environments [4]. They combine learned intrinsic rewards with exploration bonuses to enhance agent performance. Eschmann et al. has a chapter specifically on how to design the reward function for the RL agent [6].

B. Anomaly Detection in Time Series Data

For complex time series data, recently, anomaly detection algorithms have been developed. For example, 1) Skyline [7] is a real-time anomaly detection system, and 2) Twitter's anomaly detection package is designed to identify anomalies in the presence of seasonality and trends [32]. Some anomaly detection methods, such as ContextOSE [21], emphasize capturing local patterns rather than global trends. Hierarchical temporal memory (HTM), implemented in projects like Numenta, stores and recalls temporal and spatial patterns for anomaly detection [1]. With the advancement of deep learning, Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks have been widely used for predicting future values and identifying anomalies based on prediction errors. Variants of autoencoders have also been explored for anomaly detection in time series data recently [20]. Perhaps, the most relevant study to ours is RLVAL [9]. In this study, they use static rewards that cause the VAE component to dominate their reward function.

To further advance this field, a novel approach integrating the DQN algorithm with VAE, active learning, and reward shaping has been proposed in this paper. This hybrid method aims to create a more robust framework for identifying anomalies in time series data, which improves detection accuracy and adaptability.

III. Background

Before presenting our proposed method, we first provide an overview of fundamental concepts, which include 1) DQNs, 2) VAE, 3) Active Learning, and 4) reward shaping. This review ensures a clearer understanding of our approach.

A. Deep Q-Networks and Q-Learning

Q-learning learns the action-value function $Q(s, a)$ estimating expected rewards. The Q-function updates using:

$$Q_{k+1}(s,a) \leftarrow (1-\alpha)Q_k(s,a) + \alpha[R_{s,a,s'} + \gamma \max_{a'} Q_k(s',a')] \tag{1}$$

Traditional Q-learning with neural networks can be unstable [15]. DQNs address this using experience replay and target networks [16]. Experience replay stores transitions $< s, a, r, s' >$ to reduce sample correlation and improve efficiency. Target networks provide fixed references for stable updates and better convergence.

B. Variational Autoencoder

VAEs map the original feature space to latent Gaussian distributions through an encoder–decoder architecture based on neural networks. The learning objective is to maximize the intractable marginal likelihood $p(x;\theta)$, where x is a feature vector and θ represents the decoder parameters. This likelihood is approximated using the Evidence Lower Bound (ELBO) [5]:

$$\mathcal{L}(\theta,\phi;x) = \mathbb{E}_{q(z|x;\phi)}\big[\log p(x \mid z;\theta)\big] - \mathrm{KL}[q(z \mid x;\phi) \,\|\, p(z)], \tag{2}$$

where $q(z \mid x;\phi)$ is the encoder distribution approximating the posterior, and $\mathrm{KL}[\cdot \,\|\, \cdot]$ denotes the Kullback–Leibler divergence between the approximate posterior and the prior $p(z)$.

C. Active Learning in Machine Learning Systems

Active learning enhances efficiency in machine learning systems by selectively querying unlabeled data points for expert labeling. Given a labeled dataset $\mathcal{L} = (X, Y)$ and an unlabeled pool $\mathcal{U} = \{x_1, x_2, \ldots, x_n\}$, a query function Q identifies the most informative samples from $\mathcal{U}$ to refine a classifier C while minimizing the number of labeled instances required. One widely used strategy is Margin Sampling, which selects samples where the classifier is least confident. Specifically, it chooses the instance with the smallest margin between the top two predicted class probabilities:

$$x_m = \arg\min_{x \in \mathcal{U}} \Big(P_C(\hat{y}^1 \mid x) - P_C(\hat{y}^2 \mid x) \Big), \tag{3}$$

where $P_C(\hat{y}^1 \mid x)$ and $P_C(\hat{y}^2 \mid x)$ denote the first and second most probable class predictions, respectively.

D. Reward Shaping

Reward shaping modifies the reward function in RL by incorporating domain knowledge to accelerate learning when rewards are sparse or delayed. Potential-based reward shaping (PBRS) preserves the optimal policy while adding intermediate rewards to encourage desirable behaviors [12]. We extend this concept by incorporating VAE reconstruction error into the reward function, detailed in Section IV-B.

IV. Proposed Method

In this section, we describe our proposed method in detail. Our proposed method combines VAE, Deep RL, active learning, and reward shaping. Figure 1 depicts our proposed method.

A. Implementing Anomaly Detection with VAE

Because we are working with time series data, each input x can be a sliding window of length n_steps. By training the VAE on normal segments, the network learns a latent distribution that reflects normal behavior. During inference, we measure reconstruction error to detect anomalies. That is, higher than usual reconstruction errors can signal data points not explained well by the learned normal latent structure. In our implementation, this reconstruction error also factors into the RL agent's reward shaping, where the agent's action influences how it penalizes or rewards windows with high VAE reconstruction errors. Therefore, the VAE's role is as follows. 1) It acts as a learned feature extractor in the latent space, and 2) it provides a reconstruction-based anomaly score, which can guide the reinforcement learning policy to label anomalies more accurately.

In our proposed method, the VAE is constructed via two main components: the encoder and the decoder. We define a custom Sampling Layer that uses reparameterization, adding a noise term $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ scaled by $\exp\big(0.5 \cdot \log \boldsymbol{\sigma}^2\big)$. We first build a feedforward network which outputs $\boldsymbol{\mu}(\mathbf{x})$ and $\log \boldsymbol{\sigma}^2(\mathbf{x})$. Then, the Sampling Layer combines these outputs. After z is obtained, the decoder transforms it back into the original dimensionality using a few Dense layers. The loss function is computed by summing the reconstruction error (via mean squared error or cross-entropy) and the KL term.

B. Implementing Deep RL with Reward Shaping for Anomaly Detection

Our approach frames anomaly detection as a sequential decision-making task, where the Deep RL agent observes a window of time series data and it takes an action to classify the data as either normal or anomalous. The agent receives a reward signal that reflects: 1) detection accuracy, and 2) additional criteria (i.e., reconstruction error from VAE). To be more specific, at each time step t, the environment presents the agent with a state s_t, which consists of a short sliding window of the time series (for instance, n consecutive values). The agent's action a_t is to label the last point in this window. If the agent's prediction matches the true label, it gains a positive reward; otherwise, it receives a penalty, which is a negative number. Furthermore, to encourage learning more robust latent representations, a VAE-based reconstruction error is added as a bonus or penalty in the reward function, shaping the agent's behavior to pay extra attention to regions of the time series that deviate from normal patterns. Below, we explain how we defined the state, action, policy, and reward in our proposed method.

Each *state* s_t corresponds to a short sliding window of length n from the time series. Specifically, if the time series is $\{x_1, x_2, \ldots, x_T\}$, then at time t, the agent observes $\{x_{t-n+1}, \ldots, x_t\}$. This window provides a localized snapshot of recent data, which captures short-term trends or sudden deviations that might indicate anomalies. We focus on a limited history rather than the entire time series. By adopting this method, the state remains manageable in size and ensures that the agent can efficiently process each step for real-time or near real-time anomaly detection.

At each time step t, the agent takes an action $a_t \in \{0, 1\}$, where 0 indicates a prediction of normal for the last data point in the current window, and 1 indicates a prediction of anomalous. Once the action is chosen, the environment advances by one step, which shifts the sliding window forward to form the next state s_{t+1}.

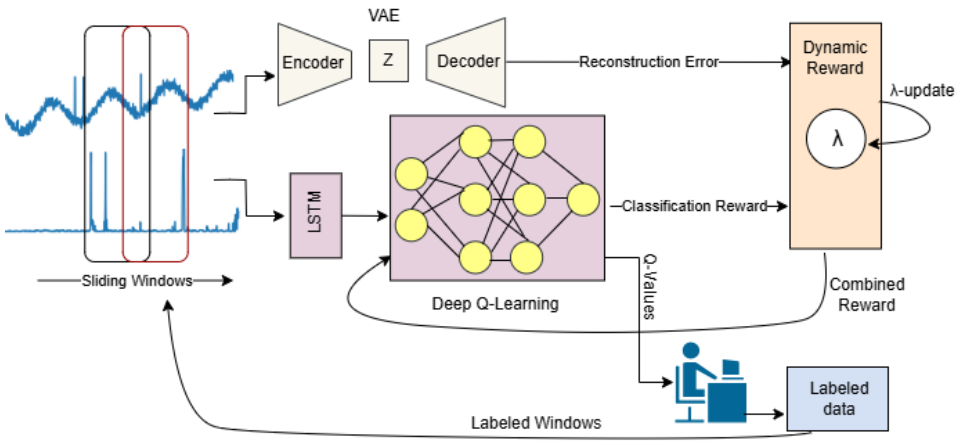


Fig. 1: Workflow of our proposed method (DRTA). Input data flows through sliding windows into two parallel components: a VAE generating reconstruction error and an LSTM-based Deep Q-Learning network for classification. The Dynamic Reward component uses an adaptive coefficient $\lambda(t)$ to balance exploration (reconstruction error R_2) and exploitation (classification rewards R_1), automatically shifting focus during training. Active learning selects uncertain samples for labeling, creating an efficient feedback loop with minimal labeled data requirements.

Next, the policy $\pi(a \mid s)$ is derived from a Q-network that estimates the action-value function $Q^\pi(s,a)$. At each time step t, given state s_t, the agent selects $a_t = \arg\max_{a\in\{0,1\}} Q^\pi(s_t, a)$. The Q-network itself is updated via the Bellman equation, which states that for an optimal policy,

$$Q^*(s,a) = \mathbb{E}\Big[r(s,a) + \gamma \max_{a'} Q^*(s',a')\Big], \tag{4}$$

where 1) $r(s,a)$ is the reward for taking action a in state s, 2) $\gamma \in (0,1)$ is the discount factor, and 3) s' is the next state. In practice, we approximate $Q^\pi(s,a)$ with a deep neural network and iteratively minimize the temporal-difference error $\big(y - Q^\pi(s_t, a_t)\big)^2$. where $y = r_t + \gamma \max_{a'} Q^\pi(s_{t+1}, a')$ is the Bellman target. As training progresses, the agent refines its Q-function. As a result, the policy that selects normal ($a = 0$) or anomalous ($a = 1$) labels for each time step is also refined.

Next, we present our reward system design for semi-supervised anomaly detection in time series data, which combines immediate classification rewards (i.e., extrinsic reward, R_1) with reconstruction-based reward shaping through VAE integration (i.e., intrinsic reward, R_2). This demonstrates superior performance in low-label systems. We define the R_1 structure as follows.

$$R_1(s_t,a_t) = \begin{cases} TP_{val} & \text{if } a_t = 1 \text{ and } y_t = 1, \\ TN_{val} & \text{if } a_t = 0 \text{ and } y_t = 0, \\ FP_{val} & \text{if } a_t = 1 \text{ and } y_t = 0, \\ FN_{val} & \text{if } a_t = 0 \text{ and } y_t = 1. \end{cases} \tag{5}$$

where, 1) $TP_{val} = 5, TN_{val} = 1$, and 2) $FP_{val} = -1, FN_{val} = -5$. This implementation creates a reward vector where index 0 represents the reward for non-anomaly classification and index 1 for anomaly classification. The asymmetric reward structure (5:1 ratio between TP and TN) reflects the higher importance of detecting true anomalies.

Next, reconstruction-based reward shaping in our proposed method uses the VAE to guide the learning process by incorporating reconstruction error as an additional reward component. The VAE is trained on normal time series data to learn a compact latent representation. This enables the VAE to reconstruct normal patterns effectively. The reconstruction error is calculated as the MSE between the original input x_t and its reconstruction $\hat{x}_t$, which serves as a measure of how well the current state aligns with normal behavior. The reconstruction error is computed as:

$$R_2(s_t,a_t) = \text{MSE}(x_t, \hat{x}_t) = \frac{1}{n}\sum_{i=1}^{n}(x_{t,i} - \hat{x}_{t,i})^2, \tag{6}$$

where n is the dimensionality of the input window.

Finally, the mathematical formulation for the total reward is:

$$R_{total}(s_t,a_t) = R_1(s_t,a_t) + \lambda(t) R_2(s_t,a_t) \tag{7}$$

where $\lambda(t)$ is a dynamic scaling coefficient that adjusts the effect of the reconstruction penalty over time. In our proposed method, the dynamic coefficient $\lambda(t)$ plays a crucial role in balancing the influence of the reconstruction error from VAE in the total reward calculation. It ensures that the agent learns to prioritize anomaly detection while still leveraging reconstruction-based guidance during training.

Reconstruction error provides an unsupervised signal that complements the supervised classification reward $R_1(s_t,a_t)$. Without scaling by $\lambda(t)$, the magnitude of the reconstruction error might dominate or be negligible compared to $R_1(s_t,a_t)$, which could lead to suboptimal learning. By dynamically adjusting $\lambda(t)$, the framework ensures that: 1) The agent explores normal patterns early in training, and 2) the focus gradually shifts toward accurate anomaly classification as training progresses.

The coefficient (i.e., $\lambda(t)$) is updated after each episode based on the total episode reward. The update rule follows a proportional control mechanism:

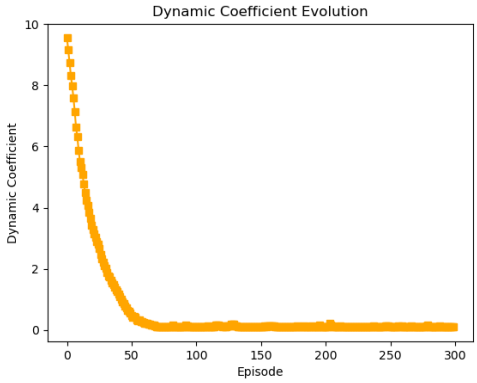
$$\lambda_{t+1} = \text{clip}\left(\lambda_t + \alpha\left(R_{\text{target}} - R_{\text{episode}}\right), \lambda_{\min}, \lambda_{\max}\right), \quad (8)$$

where: 1) R_{target}: Target reward for an episode. 2) R_{episode}: Total reward achieved in the current episode. 3) α: Learning rate for adjusting $\lambda(t)$. 4) $\lambda_{\min}$ and $\lambda_{\max}$: Minimum and maximum allowable values for $\lambda(t)$. 5) The clip function restricts a value to stay within a specified range, replacing values below $\lambda_{\min}$ with $\lambda_{\min}$ and values above $\lambda_{\max}$ with $\lambda_{\max}$.

This formula ensures that:

- If $R_{\text{episode}} < R_{\text{target}}$, then $\lambda(t)$ increases to emphasize reconstruction error.
- If $R_{\text{episode}} > R_{\text{target}}$, then $\lambda(t)$ decreases to reduce reliance on reconstruction error.

Figure 2 depicts the relationship between the dynamic coefficient evolution and the training reward during RL training. Figure 2a shows how the scaling factor $\lambda(t)$ decreases over episodes. It starts at a high value to prioritize exploration (via reconstruction error) and gradually stabilizes as the agent shifts focus to exploitation (classification accuracy). This behavior directly affects figure 2b, where initial episodes show higher rewards due to the significant contribution of reconstruction error (R_2) scaled by $\lambda(t)$. As $\lambda(t)$ decreases, the reward curve stabilizes and reflects classification performance (R_1), with fluctuations arising from variations in state-action transitions. These figures demonstrate how the dynamic reward mechanism effectively balances exploration and exploitation throughout training.



(a) Dynamic coefficient evolution over episodes

(b) Training reward curve

Fig. 2: Relationship between the dynamic coefficient and reward evolution during training.

C. Implementing Active Learning for Anomaly Detection

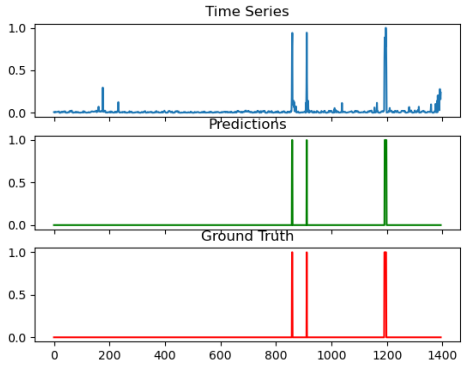
In our proposed method, the active learning module is designed to iteratively identify and label the most uncertain samples from the time series dataset to improve the agent's anomaly detection capabilities. The active learning class uses a margin-based sampling strategy. It calculates the absolute difference between the Q-values of the two possible actions (normal or anomaly) for each state (i.e., $\text{Margin}(s) = |Q(s, a_1) - Q(s, a_2)|$). First, samples with the smallest margin (i.e., the most uncertain predictions) are ranked. Then, the top-N uncertain samples are selected for manual labeling by a user (i.e., Selected Samples $= \arg\min_{s \in \mathcal{S}} \text{Margin}(s)$). This process ensures that the agent focuses on learning from the most ambiguous cases. This accelerates the agent's capability in detecting anomalies.

Once these samples are labeled, they are added back to the dataset. Then, label propagation is applied using a semi-supervised learning technique (i.e., LabelSpreading) to transmit labels to nearby unlabeled samples based on feature similarity. The probability of a label y_i for an unlabeled sample x_i is computed as:

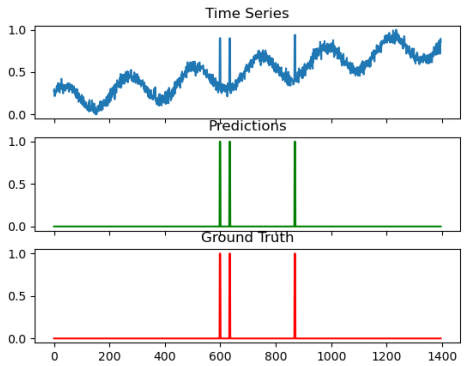
$$P(y_i|x_i) = \frac{\sum_{j \in \mathcal{L}} w_{ij} P(y_j|x_j)}{\sum_j w_{ij}} \quad (9)$$

where 1) $\mathcal{L}$ is the set of labeled samples, and 2) w_{ij} is the similarity weight between samples x_i and x_j. This combination of active learning and label propagation: 1) reduces reliance on large amounts of labeled data, and 2) enables the agent to learn effectively. The active learning process is tightly integrated into the RL loop, which allows the agent to refine its policy progressively by incorporating high-value labeled samples into its training set. Our proposed method algorithm is detailed in Algorithm 1.

Figure 3 illustrates the performance of our proposed method on two different datasets and query rates. Figure 3a represents an episode from the Yahoo A1 dataset with 1% of samples queried via active learning, which shows how well our model aligns its predictions with the ground truth anomalies. Figure 3b depicts an episode from the Yahoo A2 dataset with 10% of samples queried, which highlights the model's ability to detect anomalies effectively even with a higher query rate and a different dataset.



(a) An episode for Yahoo A1 dataset with 1% queried samples



(b) An episode for Yahoo A2 dataset with 10% queried samples

Fig. 3: Example visualizations of anomaly detection results from the proposed method.

Algorithm 1 Dynamic Reward–Scaled RL with VAE and Active Learning

Require: Normal-data path D, episodes N, batch size B, initial coefficient λ_0
Ensure: Trained Q-network; precision/recall/F1

function BUILDVAE(D, n_steps)
  Load sliding windows from D and standardize
  Train VAE on normal windows
  return `vae`, encoder
function WARMUP($env, init\_mem$)
  Fit IsolationForest on initial states; collect top-M outliers
  Play random/heuristic actions to fill $ReplayMem$
  return $ReplayMem$
function TRAINRL(env, `vae`, λ, N, B)
  Init Q-network & target network
  for $e \leftarrow 1$ **to** N **do**
    Reset env; total_reward $\leftarrow 0$
    while not done **do**
      Select action via ϵ-greedy on Q-network
      Observe $(s, r, s', done)$
      $r \leftarrow r_{\text{class}} + \lambda \times r_{\text{VAE}}$
      Append $(s, r, s', done)$ to $ReplayMem$
      Sample mini-batch of size B; update Q-network
      if step mod $K = 0$ **then**
        Sync target network
    Perform active learning: query top-k uncertain states
    $\lambda \leftarrow$ update_dynamic_coef(λ, total_reward)
function VALIDATE(env, Q-network, $episodes$)
  Run episodes with $\epsilon = 0$; accumulate precision/recall/F1
function MAIN($D, n\_steps, init\_mem, N, B$)
  `vae`, encoder $\leftarrow$ BUILDVAE(D, n_steps)
  Initialize env
  $ReplayMem \leftarrow$ WARMUP($env, init\_mem$)
  TRAINRL(env, `vae`, λ_0, N, B)
  VALIDATE(env, Q-network, $\lceil N/10 \rceil$)
  Save reward & coefficient curves

V. EXPERIMENTS

This section outlines our experiments. We first begin with dataset specifications, and then present comparative analyses against benchmark methods. We evaluate anomaly detection efficacy through three common metrics: 1) Precision measures prediction accuracy via correctly identified anomalies, 2) Recall assesses system sensitivity through true anomaly detection rates, and 3) F1-Score harmonizes both measures to mitigate evaluation bias.

A. Datasets

We evaluated our proposed method using two widely used datasets for time series anomaly detection: the Yahoo A1Benchmark and A2Benchmark. Table I summarizes the datasets' characteristics, with detailed descriptions provided in the following section.

a) Yahoo Benchmark.: This Yahoo Webscope benchmark combines real-world service metrics with synthetic examples for temporal anomaly detection. We use two subsets: A1Benchmark (67 labeled time series, 1400–1600 points each) from Yahoo login patterns and synthetic scenarios, and A2Benchmark (100 synthetic time series with single outlier anomalies). Both provide precise anomaly annotations for strict method evaluation.

TABLE I: Overview of datsets

dataset	total points	anomalies
Yahoo A1	94866	1669
Yahoo A2	142100	400

B. Results and Discussions

In this section, we compare our proposed method to the best studies in the literature. These studies are as follows. 1) Luminol, which is a lightweight Python library that uses various statistical anomaly detection algorithms to directly return anomaly scores without requiring training [17]. 2) SPOT, which detects anomalies in streaming univariate time series by dynamically selecting thresholds based on extreme value theory. 3) DSPOT extends SPOT by adapting thresholds dynamically to changes in data distribution [36]. 4) SR-CNN, which generates synthetic training data with injected fake anomalies and applies Spectral Residual (SR) methods to train its neural network [27]. 5) DAGMM combines a deep

autoencoder for low-dimensional representation with a Gaussian Mixture Model (GMM) to jointly optimize reconstruction error and probabilistic modeling [37]. 6) Autoencoder uses replicator neural networks with three hidden layers to measure the outlyingness of data records based on reconstruction error [11]. 7) Deep SAD learns a latent distribution of normal data with low entropy while ensuring that anomalies have higher entropy distributions [24]. 8) RLAD, which combines RL and active learning to detect anomalies [34]. 9) RLVAL, which improves RLAD by using VAE [9].

The results in Table II and Table III demonstrate the effectiveness of our proposed method in anomaly detection across the Yahoo A1 and Yahoo A2 datasets, in particular when compared to both unsupervised and semi-supervised baselines. Among the unsupervised methods, SPOT and DSPOT achieve relatively higher F1-scores (e.g., 0.446 and 0.442 on Yahoo A1, respectively). However, their performance is significantly lower than semi-supervised methods that leverage labeled data. Semi-supervised methods such as Deep SAD and RLAD show improved performance as the number of queried samples increases (from 1% to 5% to 10% of total data samples), with RLVAL achieving an F1-score of 0.921 on Yahoo A1 when querying 10% of samples. However, our proposed method outperforms all baselines at lower query rates, which achieves an F1-score of 0.90 on Yahoo A1 and 0.80 on Yahoo A2 with only 1% queried samples. This demonstrates our proposed method's robustness in low-label systems.

At 5% queried samples, our method achieves an F1-score of 0.888 on both Yahoo A1 and Yahoo A2 datasets, maintaining competitive performance compared to RLVAL (F1-score: 0.872 on Yahoo A1). This trend continues at 10%, where our method achieves consistent results (F1-score: 0.666), while RLVAL outperforms on Yahoo A1 (F1-score: 0.921). Notably, RLVAL does not report results for Yahoo A2, making it difficult to assess its generalizability across datasets. In contrast, our method demonstrates strong performance across both datasets, which highlights its adaptability.

It is crucial to note that the adaptive reward mechanism in our proposed method plays an important role in reducing the need for labeled data while maintaining high anomaly detection performance. At 1% queried samples, the dynamic reward shaping mechanism effectively balances exploration (via reconstruction error, R_2) and exploitation (via classification rewards, R_1), which enables the model to achieve high recall (1.00 on Yahoo A2 and 0.90 on Yahoo A1) while maintaining precision. This balance allows our method to outperform RLVAL at this query rate (F1-score: 0.834 on Yahoo A1), which demonstrates its ability to generalize from minimal supervision.

As the number of queried samples increases to 5% and 10%, the adaptive reward mechanism shifts focus toward classification accuracy by dynamically adjusting the contribution of reconstruction error through $\lambda(t)$. This ensures that precision improves without sacrificing recall, enabling consistent performance across datasets with minimal dependency on labeled data. The integration of active learning further improves this efficiency by focusing labeling efforts on the most uncertain samples, which ensures that limited labeling resources are utilized effectively.

In summary, our proposed method achieves superior performance across various query rates by using adaptive rewards and active learning to reduce labeling requirements while maintaining robust anomaly detection capabilities across diverse datasets.

TABLE II: Performance comparison on Yahoo A1

Models	F1-score	Precision	Recall
Unsupervised			
luminol	0.177	0.261	0.258
SR-CNN	0.254	0.408	0.382
SPOT	0.446	0.513	0.394
DSPOT	0.442	0.512	0.398
DAGMM	0.295	0.317	0.309
Autoencoder	0.438	0.471	0.431
Semi-supervised			
Deep SAD (1%)	0.594	0.518	0.689
Deep SAD (5%)	0.671	0.603	0.681
Deep SAD (10%)	0.727	0.672	0.758
RLAD (1%)	0.708	0.652	0.652
RLAD (5%)	0.752	0.710	0.800
RLAD (10%)	0.797	0.733	0.922
RLVAL (1%)	0.834	0.819	0.850
RLVAL (5%)	0.872	0.846	0.900
RLVAL (10%)	**0.921**	0.894	0.950
Proposed Method			
DRTA (1%)	**0.900**	0.900	0.900
DRTA (5%)	**0.888**	1.000	0.800
DRTA (10%)	0.666	1.000	0.500

TABLE III: Performance comparison on Yahoo A2

Models	F1-score	Precision	Recall
Unsupervised			
luminol	0.277	0.314	0.266
SR-CNN	0.374	0.408	0.346
SPOT	0.691	0.681	0.646
DSPOT	0.595	0.625	0.467
DAGMM	0.455	0.458	0.459
Autoencoder	0.465	0.484	0.435
Semi-supervised			
Deep SAD (1%)	0.754	0.712	0.799
Deep SAD (5%)	0.825	0.801	0.854
Deep SAD (10%)	0.851	0.832	0.879
RLAD (1%)	**1.000**	1.000	1.000
RLAD (5%)	**1.000**	1.000	1.000
RLAD (10%)	1.000	1.000	1.000
Proposed Method			
DRTA (1%)	0.800	0.666	1.000
DRTA (5%)	0.888	0.800	1.000
DRTA (10%)	**1.000**	1.000	1.000

VI. Conclusion

In this study, we proposed a novel RL-based framework for time series anomaly detection which integrates dynamic reward shaping, VAE, and active learning. Our method effectively balances exploration and exploitation by dynamically scaling reconstruction error contributions through the adaptive coefficient. This enables the agent to learn robust anomaly detection policies even in low-label systems. The integration of active learning further improves efficiency by focusing labeling efforts on the most uncertain samples. This significantly reduces the dependency on large labeled datasets. Experimental results on the Yahoo A1 and A2 datasets demonstrate that our method consistently outperforms state-of-the-art approaches. For future work, we suggest exploring the integration of large language models (LLMs) into our framework to enhance interpretability and decision-making in anomaly detection.

References

[1] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.

[2] F. Bourdonnaye, C. Teulière, T. Chateau, and J. Triesch, "Learning of binocular fixations using anomaly detection with deep reinforcement learning," in *Proc. Int. Joint Conf. on Neural Networks*, pp. 760–767, 2017.

[3] K. Das and J. Schneider, "Detecting anomalous records in categorical datasets," in *Proc. Int. Conf. on Knowledge Discovery and Data Mining*, pp. 220–229, 2007.

[4] R. Devidze, P. Kamalaruban, and A. Singla, "Exploration-Guided Reward Shaping for Reinforcement Learning under Sparse Rewards," *36th Conf. on Neural Information Processing Systems (NeurIPS 2022)*, 2022.

[5] E. A. Elaziz, R. Fathalla, and M. Shaheen, "Deep Reinforcement Learning for Data-Efficient Weakly Supervised Business Process Anomaly Detection," *Journal of Big Data*, vol. 10, no. 1, p. 33, Mar. 2023. Available: https://doi.org/10.1186/s40537-023-00708-5.

[6] J. Eschmann, "Reward Function Design in Reinforcement Learning," in *Reinforcement Learning Algorithms: Analysis and Applications*, B. Belousov, H. Abdulsamad, P. Klink, S. Parisi, and J. Peters, Eds., Studies in Computational Intelligence, vol. 883, Springer, 2021. doi: 10.1007/978-3-030-70441-3_4.

[7] Esty, "Skyline," 2014. [Online]. Available: https://github.com/etsy/skyline.

[8] R. Fontugne, J. Ortiz, N. Tremblay, P. Borgnat, P. Flandrin, K. Fukuda, D. Culler, and H. Esaki, "Strip, Bind, and Search: A Method for Identifying Abnormal Energy Consumption in Buildings," in *Proc. 12th Int. Conf. on Information Processing in Sensor Networks (IPSN '13)*, Philadelphia, PA, USA, pp. 129–140, 2013. doi: 10.1145/2461381.2461399.

[9] B. Golchin and B. Rekabdar, "Anomaly Detection in Time Series Data Using Reinforcement Learning, Variational Autoencoder, and Active Learning," *2024 Conf. on AI, Science, Engineering, and Technology (AIxSET)*, pp. 1–8, 2024.

[10] B. Golchin and N. Riahi, "Emotion Detection in Twitter Messages Using Combination of Long Short-Term Memory and Convolutional Deep Neural Networks," *Int. J. Comput. Inf. Eng.*, vol. 15, pp. 578–585, 2021. doi: 10.1145/1824766.1824773.

[11] S. Hawkins, H. He, G. Williams, and R. Baxter, "Outlier Detection Using Replicator Neural Networks," in *Proc. Int. Conf. on Data Warehousing and Knowledge Discovery*, pp. 170–180, 2002. Springer.

[12] I. Hong, Z. Li, A. Bukharin, Y. Li, H. Jiang, T. Yang, and T. Zhao, "Adaptive Preference Scaling for Reinforcement Learning with Human Feedback," in *Advances in Neural Information Processing Systems*, vol. 37, pp. 107249–107269, 2025.

[13] C. Huang, Y. Wu, Y. Zuo, K. Pei, and G. Min, "Towards experienced anomaly detector through reinforcement learning," in *Proc. AAAI Conf. on Artificial Intelligence*, pp. 8087–8088, 2018.

[14] P. Krajsic and B. Franczyk, "Semi-Supervised Anomaly Detection in Business Process Event Data Using Self-Attention Based Classification," in *Procedia Computer Science*, Proc. 25th Int. Conf. on Knowledge-Based and Intelligent Information and Engineering Systems, pp. 39–48, 2021.

[15] L. Li, "Deep reinforcement learning: An overview," *arXiv preprint*, arXiv:1701.07274, 2017.

[16] L. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Machine Learning*, vol. 8, pp. 293–321, 1992.

[17] LinkedIn, "LinkedIn/luminol," [Online]. Available: https://github.com/linkedin/luminol. Accessed: 2019-04-24.

[18] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in *Proc. IEEE Int. Conf. on Data Mining (ICDM)*, 2008.

[19] J. Ma and S. Perkins, "Time-series novelty detection using one-class support vector machines," in *Proc. Int. Joint Conf. on Neural Networks*, vol. 3, pp. 1741–1745, 2003.

[20] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "LSTM-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint*, arXiv:1607.00148, pp. 1–5, 2016.

[21] S. Mikhail, "Contextual Anomaly Detection," 2015. [Online]. Available: https://github.com/smirmik/CAD.

[22] H. Ma, Z. Luo, T. V. Vo, K. Sima, and T.-Y. Leong, "Highly Efficient Self-Adaptive Reward Shaping for Reinforcement Learning," 2024.

[23] S. Mukkamala, G. Janoski, and A. Sung, "Intrusion detection using neural networks and support vector machines," in *Proc. Int. Joint Conf. on Neural Networks*, vol. 2, pp. 1702–1707, 2002.

[24] G. Pang, C. Shen, and A. van den Hengel, "Deep Anomaly Detection with Deviation Networks," in *Proc. 25th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*, pp. 353–362, 2019. ACM.

[25] I. C. Paschalidis and Y. Chen, "Statistical anomaly detection with sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 7, no. 2, p. 23, Sep. 2010.

[26] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *Proc. ACM SIGMOD Record*, vol. 29, pp. 427–438, 2000.

[27] H. Ren, B. Xu, Y. Wang, C. Yi, C. Huang, X. Kou, T. Xing, M. Yang, J. Tong, and Q. Zhang, "Time-Series Anomaly Detection Service at Microsoft," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*, Anchorage, AK, USA, pp. 3009–3017, 2019. doi: 10.1145/3292500.3330680.

[28] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge: MIT Press, 1998.

[29] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy Gradient Methods for Reinforcement Learning with Function Approximation," in *Advances in Neural Information Processing Systems 12*, pp. 1057–1063, 2000.

[30] G. Tandon and P. K. Chan, "Weighting versus pruning in rule validation for detecting network and host anomalies," in *Proc. Int. Conf. on Knowledge Discovery and Data Mining*, pp. 697–706, 2007.

[31] J. G. Thomas, S. P. Mudur, and N. Shiri, "Detecting Anomalous Behaviour from Textual Content in Financial Records," in *Proc. IEEE/WIC/ACM Int. Conf. on Web Intelligence (WI '19)*, Thessaloniki, Greece, pp. 373–377, 2019. doi: 10.1145/3350546.3352550.

[32] Twitter, "Twitter Anomaly Detection," 2015. [Online]. Available: https://github.com/twitter/AnomalyDetection/releases.

[33] R. J. Williams, "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning," *Machine Learning*, vol. 8, pp. 229–256, 1992.

[34] T. Wu and J. Ortiz, "RLAD: Time Series Anomaly Detection through Reinforcement Learning and Active Learning," arXiv preprint, arXiv:2104.00543, Mar. 2021. Available: http://arxiv.org/abs/2104.00543.

[35] T. Zhu, Y. Guo, J. Ma, and A. Ju, "Business Process Mining Based Insider Threat Detection System," in *Advances on P2P, Parallel, Grid, Cloud and Internet Computing*, F. Xhafa, L. Barolli, and F. Amato, Eds., Lecture Notes on Data Engineering and Communications Technologies, vol. 1, Cham: Springer, pp. 467–478, 2017.

[36] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," Tech. Rep. CMU-CALD-02–107, Carnegie Mellon University, 2002.

[37] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection," 2018. (Unpublished manuscript).