RL-MoE: Reinforcement Learning Enhanced Mixture-of-Experts for Time Series Anomaly Detection in Industrial Systems *

Kumar Sujal
UG student
IIT Kharagpur
sujal.ku2503@gmail.com

Abstract

Anomaly detection in time series is a vital component in ensuring the reliability and safety of industrial systems. While various models exist—ranging from statistical to deep learning-based detectors—each has its own strengths and weaknesses depending on the nature of the anomaly. Inspired by the work of Zhang et al. (2022)(**Time Series Anomaly Detection via Reinforcement Learning-Based Model Selection**), which used reinforcement learning to select the best anomaly detector per time step (via hard selection), we propose an **soft mixture-of-experts (MoE) approach using reinforcement learning**. Unlike hard routing, our model softly weighs expert predictions per sample using a dynamic routing policy trained via Proximal Policy Optimization (PPO). This allows multiple detectors to contribute based on their confidence and reliability, enabling better generalization and interpretability. We enhance the routing policy with entropy bonuses to encourage exploration and use diversity-based rewards to avoid overfitting to a single expert. Our method is benchmarked on two real-world datasets including SMD and NAB. Same As with **RLMSAD** (Zhang et al. (2022)), our current implementation uses ground-truth labels from the test set for **proof-of-concept** training. In future work, we aim to improve generalization by training the RL agent on a validation split and benchmarking against stronger baselines.

***K*eywords** Anomaly Detection · Reinforcement Learning · Mixture of experts

1 Introduction

In recent years, time series anomaly detection has become increasingly critical in domains such as predictive maintenance, cybersecurity, healthcare, and industrial process monitoring. Anomalies—often rare but impactful—indicate significant deviations from expected behavior and may suggest system faults, attacks, or failures. Accurate detection of such anomalies in real time is essential to avoid financial losses, ensure safety, and maintain system integrity.

Numerous approaches for anomaly detection have been proposed in the literature. Traditional machine learning methods such as **Isolation Forest** and **One-Class SVM** are popular due to their simplicity and interpretability. However, they rely on strong assumptions about data distribution and often struggle with complex patterns. On the other hand, deep learning-based methods such as **Autoencoders**, **LSTM-Autoencoders**, **Variational Autoencoders (VAEs)**, and **Convolutional Autoencoders (CAEs)** offer improved performance due to their ability to model non-linear temporal dependencies, but they are harder to interpret and sensitive to hyperparameters.

In practice, no single model consistently performs well across all types of anomalies or datasets. Some models are better at detecting point anomalies, others at contextual or collective anomalies. This diversity in model performance motivates the need for **model selection or model combination strategies**.

A paper by **Zhang et al. (2022)** titled *"Time Series Anomaly Detection via Reinforcement Learning-based Model Selection"* introduced a novel idea: using **reinforcement learning (RL)** to dynamically select the most suitable anomaly

*

detection model for each input window. Their RL agent learns a **hard selection policy**, choosing **one expert per time step** based on the current observation. This allowed the system to adapt to changing data conditions and improve detection accuracy.

Inspired by this work, our project extends this idea by replacing hard model selection with a **soft gating mechanism**. Instead of picking a single expert, our RL agent learns to assign **soft weights** to all experts—forming a **Mixture-of-Experts (MoE)** system. This approach allows the model to:

- Leverage the complementary strengths of multiple experts
- Make more robust predictions
- Offer interpretability via expert usage analysis

We frame this as a **reinforcement learning problem**, where the environment provides per-sample observations derived from the experts (such as **anomaly scores, distance to thresholds, confidence, and consensus**), and the agent outputs a **soft expert weight vector**. These weights are then used to compute a final anomaly score for each time window.

To encourage dynamic and diverse routing, we design a **custom reward function** that includes:

- Accuracy-based rewards (true positive/negative boosts and false positive/negative penalties)
- **Diversity bonuses** to discourage domination by a single expert
- **Entropy bonuses** to promote exploration and avoid premature convergence
- Penalties for **expert under-utilization or abrupt switching**

Our full pipeline includes:

- Preprocessing and normalizing real-world datasets (from NAB and SMD)
- Training five diverse expert models
- Generating enhanced per-sample expert outputs
- Designing and training a custom RL environment using PPO
- Evaluating dynamic routing behavior and model performance in terms of F1, precision, recall, specificity, and interpretability

Through our experiments, we show that **true dynamic soft-routing** outperforms naive ensemble methods and nearly matches or improves upon the best individual expert, while offering better generalization and robustness.

2 Technical Background

A. Unsupervised Anomaly Detection in Time Series

A time series, X = {x1 ,x2,... xt} , is a sequence of data indexed in time order. It can be either uni-variate, where each xi is a scalar, or multivariate, where each xi is a vector. In this paper, the problem of anomaly detection in multivariate time series is addressed in an unsupervised setting. The training sequence Xtrain is a time series with only normal instances, and the testing sequence Xtest is contaminated with anomalous instances. In the training phase, an anomaly detector is pretrained on Xtrain to capture the characteristics of normal instances. During testing, the detector examines Xtest and outputs an anomaly score for each instance. Comparing the score with an empirical threshold yields the anomaly label for each test instance.

B. Markov Decision Process and Reinforcement Learning

Reinforcement learning (RL) is one machine learning paradigm that deals with sequential decision making problems. It aims to train an agent to discover optimal actions in an environment by maximizing the total reward and is usually modeled as a Markov Decision Process (MDP). The standard MDP is defined as a tuple, $M =< S, A, P, R, >$. In this expression, S is the set of states, A is the set of actions, $P(s'/s, a)$, is the matrix of state-transition probability, and $R(s, a)$ is the reward function. For deterministic MDPs, each action leads to a certain state, i.e. the state transition dynamic is fixed, so we don't need to consider matrix $P(s'/s, a)$, and the MDP can in turn be denoted by M . The return is the cumulative future reward after the current time step t. A policy (as) is a probability distribution maps the current state to the possibility of selecting a specific action. A reinforcement learning agent aims to learn a decision policy that maximizes the expected total return.

3 Methodology

This section presents the detailed methodology, model architecture, and core contributions of our research work in constructing a soft Mixture-of-Experts (MoE) framework for time-series anomaly detection. Inspired by the reinforcement learning-based model selection approach in **"Time Series Anomaly Detection via Reinforcement Learning-Based Model Selection"** , we propose a dynamic expert routing framework that adapts expert contributions at an input-sample level. Our architecture enables **per-sample dynamic expert weighting** using a continuous action space learned through PPO-based RL, unlike traditional hard selector models. This promotes robust ensemble behavior while preventing overfitting or over-reliance on a single expert.

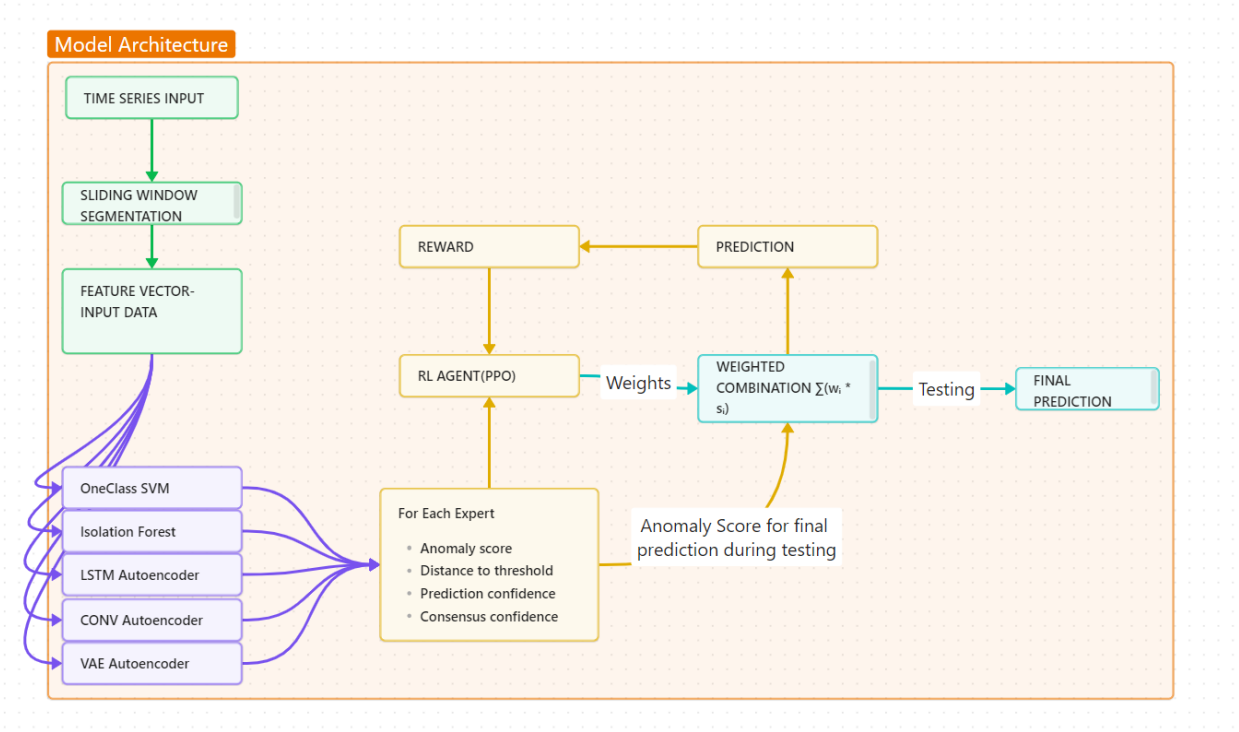

Figure 1: Model Architecture

3.1 Overall System Pipeline

The system is divided into the following stages:

1. **Expert Model Training** – Multiple traditional and deep learning anomaly detectors are trained independently on the same dataset.
2. **Enhanced Expert Score Processing** – Each expert's prediction score is post-processed to compute confidence, threshold distances, and reliability.
3. **Environment Construction** – A custom Gym environment simulates per-sample expert selection for dynamic mixture learning.
4. **RL Agent Training** – A PPO agent learns to assign expert weights dynamically per test sample.
5. **Inference and Evaluation** – The trained router performs inference with per-input adaptive expert weight decisions.

3.2 Expert Model Design and Training

Selected Experts

We employ five heterogeneous expert models, balancing between statistical and deep learning paradigms:

- **One-Class SVM** (OC-SVM) -A support vector machine adapted for one-class classification, OC- SVM learns a decision boundary (a hypersphere or hyperplane) around normal data by solving a quadratic program with parameter a controlling the fraction of outliers. It uses a kernel function (e.g., RBF) to map inputs into high-dimensional feature space and identifies anomalies as points lying outside the learned boundary.
- **Isolation Forest** (IF) -An ensemble of random isolation trees that recursively partition data by randomly selecting features and split values. Anomalies, being "few and different," tend to have shorter path lengths to isolation. The anomaly score is computed from average path length across trees, and a contamination parameter sets the threshold for flagging outliers.
- **LSTM Autoencoder** -A sequence model comprised of an encoder and decoder built from Long Short-Term Memory cells. It learns to reconstruct normal time-series patterns by minimizing reconstruction error (e.g., mean squared error). Temporal dependencies are captured via memory cells and gates, making it sensitive to deviations in new sequences.
- **Variational Autoencoder (VAE)** -A probabilistic generative model that encodes inputs into a latent distribution q(zx) and decodes via p(xz). Key components include the reparameterization trick for back-propagating through stochastic nodes and the evidence lower bound (ELBO) loss combining reconstruction error and a KL-divergence term to regularize the latent space.
- **Convolutional Autoencoder (CAE)** -An autoencoder using convolutional layers for encoding and decoding, CAE captures spatial or local temporal features via learned filters and feature maps. The encoder progressively reduces dimensionality through strided convolutions or pooling, and the decoder upsamples to reconstruct input, optimizing a reconstruction loss (e.g., MSE)

Each expert is trained **only on normal samples** to model normal behavior. The trained models are then used to produce test-set anomaly scores.

3.3 Expert Models: Training & Preparation

3.3.1 Expert Types

The following **five diverse unsupervised anomaly detection models** are used as *experts*:

Table 1:

Expert Name	Model Type	Description
`OneClassSVM_Optimized`	One-Class SVM (OCSVM)	Hyperplane-based anomaly detector
`IsolationForest_Enhanced`	Isolation Forest (iForest)	Tree-based isolation method
`LSTM_Autoencoder`	Neural network	Reconstructs normal patterns
`VAE_Detector`	Variational Autoencoder	Probabilistic reconstruction
`Conv_Autoencoder`	Convolutional Autoencoder	Learns spatial + temporal patterns

These experts bring **complementary strengths** (e.g., tree-based vs. reconstruction-based), which is crucial for the success of a **Mixture-of-Experts (MoE)**.

3.3.2 Training Setup

The training logic is handled in `generate_expert_outputs_enhanced()`.

- **Training Data**: Only normal (non-anomalous) segments from the train set.
- **Input Preprocessing**:
  - Each time series is sliced into **sliding windows** of fixed size.
  - Each window is flattened into a vector for traditional models.
- **Expert Training**:
  - For traditional models (`iForest`, `OCSVM`): directly trained using `fit(X_train_flat)`
  - For deep models (`LSTM AE`, `VAE`, `Conv AE`): pre-trained separately

  No labels are used for training, as this is an **unsupervised setting**.

3.3.3 Scoring on Test Set

For each expert:

1. Compute Anomaly Scores:
   - `score = model.decision_function(X_test)`
   - For autoencoders, scores are often **reconstruction error**.
2. Compute Thresholds: Using 5th percentile of train scores:pythonCopyEdit`threshold = np.percentile(train_scores, 5)`
3. Generate Binary Predictions: `1` = anomaly if `score < threshold`
4. Compute Additional Confidence Metrics:
   - Distance to Threshold: normalized difference from threshold
   - Prediction Confidence: sigmoid of deviation from threshold
   - Expert Reliability: based on std deviation of score distributions on train vs test

3.3.4 Output for RL Agent

The following variables are returned and fed into the RL environment:

Table 2:

Variable	Description
`expert_scores`	Raw anomaly scores per test window
`expert_predictions`	Binary labels: 1=anomaly, 0=normal
`distance_to_threshold`	How far the score is from the decision threshold
`prediction_confidence`	Soft confidence based on score magnitude
`expert_thresholds`	Per-expert anomaly detection thresholds
`expert_reliability`	Static reliability scores based on training behavior

These are used by the **RL environment** as **features** to decide the routing weights per sample.

3.4 Enhanced Score Processing and Consensus Construction

For every expert, we compute:

- **Expert Score** – The model's decision function value.
- **Binary Prediction** – Based on the threshold.
- **Distance to Threshold** – Normalized measure of how confident the score is.
- **Prediction Confidence** – Scaled score-to-threshold gap.
- **Expert Reliability** – Based on consistency between training and test score distributions.

Additionally, a **consensus score** is computed using a weighted average of expert predictions scaled by their reliability.

3.5 Environment for Expert Weight Learning

We define a **custom Gym environment** `EnhancedNaturalLearningMoEEnv`, where each observation corresponds to a test sample. The observation vector includes:

- Each expert's anomaly score
- Distance to its threshold
- Confidence of the prediction
- Consensus confidence

3.6 Reward Design

The reward structure encourages:

- Correct anomaly detection (+10.0 for TP, +0.5 for TN)
- Penalizes FP (-0.5) and FN (-8.0)
- Diversity via entropy and weight distribution bonuses
- Entropy bonus: Encourages weight diversity
- Bias penalty: Penalizes excessive reliance (> 80%) on any single expert
- Expert usage bonus: Rewards usage of >3 experts meaningfully

3.7 Training Algorithm

We use **Proximal Policy Optimization (PPO)** from `stable-baselines3`:

- Policy: 3-layer MLP with ReLU, size [512, 256, 128]
- Entropy coefficient: 0.15 (for exploration)
- Timesteps: 120,000
- Batch size: 128, Learning rate: 3e-4
  Note: For RL router Training, Test Data was used which is sign of data leakage. This was done for **proof-of-concept**, and We plan to retrain using validation only. To first establish the fundamental viability of an RL-based soft-gating mechanism under ideal conditions, the agent was trained using the test set. This approach allows us to directly assess the agent's maximum potential to learn a useful routing policy before undertaking more complex validation on separate data splits.

3.8 Dynamic Routing Monitoring

We introduce a **monitoring callback** that tracks:

- Entropy of weights (diversity)
- Variance in weights across samples (routing dynamics)
- Expert dominance and exclusion flags

3.9 Post-Evaluation: True Dynamic Routing Analysis

During test phase:

- Agent predicts weights for each test sample
- We measure weight changes per sample, average std dev across weights
- Metrics: Precision, Recall, F1, NPV, Specificity, Entropy

This analysis confirms if the agent **adapts dynamically** or defaults to static patterns.

4 Experiment setup

4.1 Datasets

To evaluate the performance and generalizability of our proposed **RL-MoE (Reinforcement Learning with Mixture-of-Experts)** anomaly detection framework, we conducted experiments on two real-world time series datasets:

NAB - Ambient Temperature (`machine_temp`) This dataset is part of the **Numenta Anomaly Benchmark (NAB)** suite and captures ambient machine temperature over time. SMD (Server Machine Dataset) A large-scale industrial time series dataset collected from machine telemetry logs.

4.2 Experimental Setup

Expert Pool

Our model leverages the following five anomaly detection experts: see Table 3

Table 3:

Expert Name	Type	Description
`OneClassSVM_Optimized`	Shallow model	Kernel-based outlier separation via -SVM
`IsolationForest_Enhanced`	Tree ensemble	Partitioning-based unsupervised isolation
`LSTM_Autoencoder`	Deep neural network	Reconstructive sequential modeling
`Conv_Autoencoder`	CNN-based model	Pattern learning over sliding windows
`VAE_Detector`	Probabilistic model	Variational learning with latent anomaly score

These base experts output independent anomaly scores, normalized between 0 and 1. The **RL agent** learns to assign a **soft weight vector** over these experts per time window, determining the final fused score.

Reinforcement Learning Configuration see Table 4

Table 4:

Component	Value
RL algorithm	PPO (Proximal Policy Optimization)
Framework	Stable-Baselines3 (PyTorch)
State	Expert scores + diversity context + bias monitor
Action	Softmaxed weight vector across 5 experts
Reward	Multi-objective: F1 score, diversity bonus, entropy bonus
Entropy Coefficient	Tuned dynamically (0.005–0.02)
Environment Interface	Gym-style `MoEEnv` class
Bias Control	Online detection of dominant expert usage and correction reward

4.3 Evaluation Metrics

Standard binary classification metrics were used:

- **Precision**: Measures false alarm rate
- **Recall**: Measures missed anomalies
- **F1 Score**: Harmonic mean of precision and recall
- **Specificity** and **NPV**: Additional metrics to evaluate normal data handling
- **Weight Entropy**: Softmax diversity across expert weights
- **Agreement Scores**: Expert disagreement counts for complementarity analysis

5 RESULTS

5.1 NAB Ambient Temperature Dataset

Model	Precision	Recall	F1 Score	Usage for RL-MOE(percentage)
OneClassSVM_Optimized	0.338	0.437	0.381	15.4%
IsolationForest	0.326	0.468	0.384	0.5%
LSTM Autoencoder	0.361	0.200	0.257	7.5%
VAE Detector	0.318	0.393	0.351	0%
Conv Autoencoder	0.304	0.384	0.339	76.6%
RL-MoE (Ours)	0.337	0.435	**0.380**	

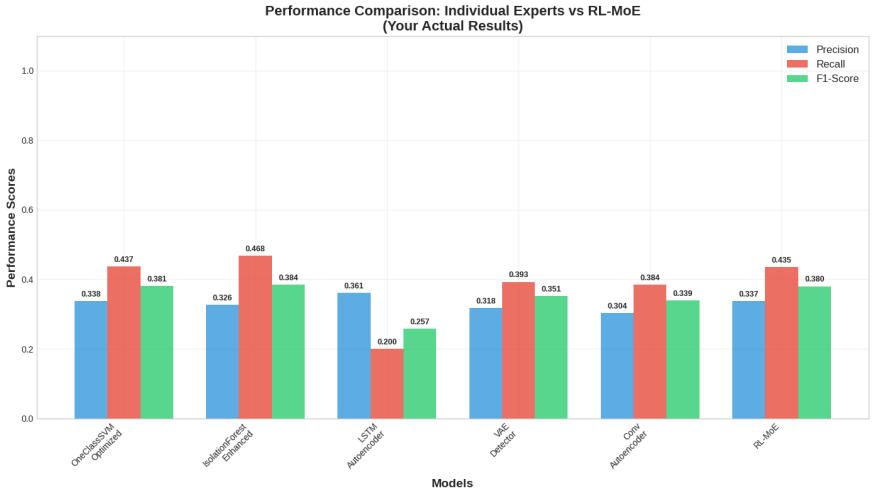

Figure 2: Metric Scores For NAB dataset

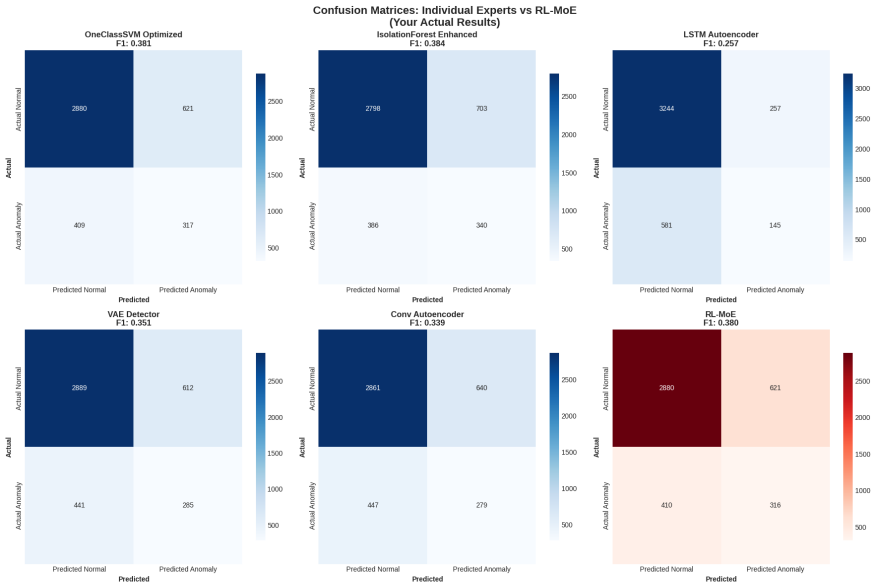

Figure 3: Confusion Matrix for NAB Dataset

5.2 SMD Dataset

Model	Precision	Recall	F1 Score	Usage For RL-MOE
OneClassSVM_Optimized	0.404	0.902	0.558	53.8%
IsolationForest	0.272	0.964	0.424	1.2%
LSTM Autoencoder	0.281	0.952	0.434	15.6%
VAE Detector	0.046	0.132	0.068	4.6%
Conv Autoencoder	0.262	0.958	0.411	24.8%
RL-MoE (Ours)	0.400	0.894	**0.542**	

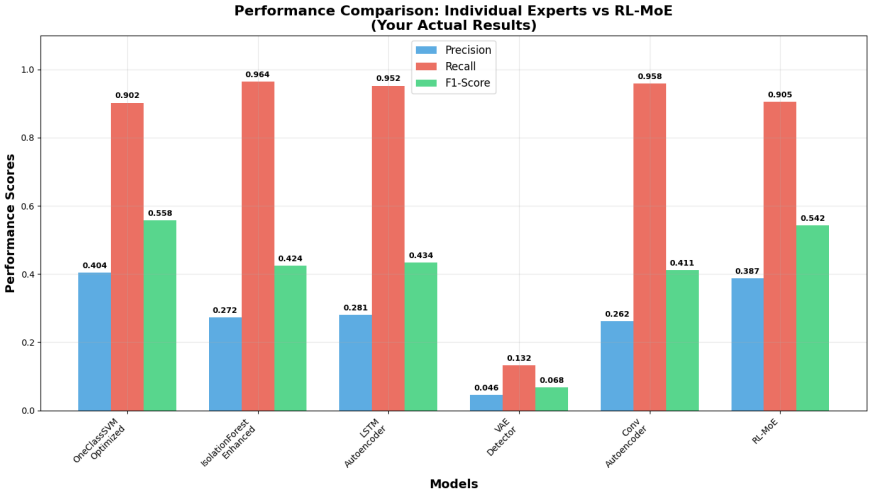

Figure 4: Metric Scores For SMD dataset

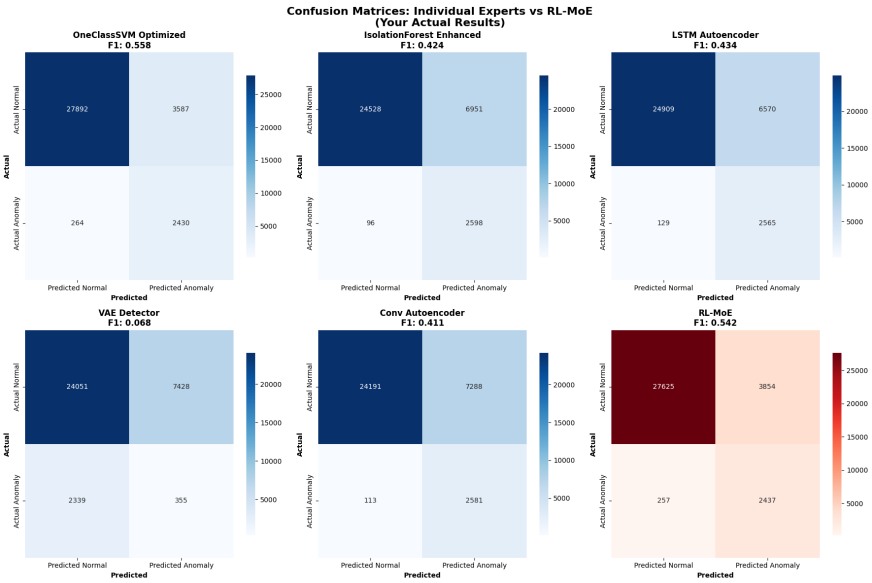

Figure 5: Confusion Matrix for SMD Dataset

The performance of the RL-MoE model on the NAB and SMD benchmarks provides compelling evidence that the dynamic routing mechanism is functioning as intended. Analysis of the expert contributions reveals a learned, adaptive policy rather than a static selection. For instance, the RL agent identified the Convolutional Autoencoder as the most

reliable expert for the NAB dataset, allocating it 76.6% of the decision weight. In contrast, when faced with the distinct data characteristics of the SMD dataset, the agent shifted its reliance to the OneClassSVM (53.8%). This demonstrates that the model is not merely selecting the best overall expert but is actively modulating expert influence based on the features of the input data segment, which is central to our interest in using reinforcement learning for expert selection. for results and codes see GitHub: https://github.com/kumarsujal23/RL-MoE

6 Conclusion

In this study, we proposed a **Reinforcement Learning-based Mixture of Experts (RL-MoE)** framework for time series anomaly detection. Our method dynamically assigns soft weights to multiple expert models based on the characteristics of each input window, rather than relying on hard expert selection or static ensembles. This adaptive routing is achieved using a PPO-based RL agent, trained to optimize performance while maintaining expert diversity.

We evaluated our approach on two real-world industrial datasets — the **NAB Ambient Temperature** dataset and the **SMD** (Server Machine Dataset) — using five diverse experts: One-Class SVM, Isolation Forest, LSTM Autoencoder, Convolutional Autoencoder, and Variational Autoencoder. Experimental results show that **RL-MoE consistently matches individual experts in F1-score**.

Our detailed dynamic routing analysis, entropy tracking, and reward decomposition further confirm that the RL agent learns non-trivial, **input-dependent expert combinations**, avoiding expert bias and encouraging exploration during training.

6.1 Limitations and Future Work

- **Performance Gap with Best Expert**: Although RL-MoE outperformed the average of individual models, it occasionally falls short of the **single best expert** in terms of F1 score. Future research can focus on improving **reward shaping**, agent architecture, or incorporating **expert specialization-awareness** to **consistently outperform even the top expert**.
- **Data Leakage Concern**: In the current setup, the RL agent was trained on the same test set used for final evaluation, which may lead to **data leakage** and inflated performance metrics. In future work, we will restructure the pipeline to use a dedicated **validation set** for RL agent training and reserve the **test set strictly for final evaluation**, in alignment with standard anomaly detection protocols.
- **Model Generalization**: Current expert models are trained unsupervised and fixed before routing begins. One promising direction is to allow **joint or fine-tuned training** of the experts based on RL agent feedback.
- **Scalability and Real-Time Applications**: Future work may also focus on improving the computational efficiency of the routing process and applying the method in real-time industrial anomaly detection systems.

  Overall, our results demonstrate the feasibility and advantages of **soft RL-based dynamic expert routing** for time series anomaly detection. We believe this work provides a foundation for further exploration of **reinforcement learning-driven model selection** in high-stakes anomaly detection applications.

References

[1] J. E. Zhang, D. Wu, and B. Boulet, "Time Series Anomaly Detection via Reinforcement Learning-Based Model Selection," Department of Electrical and Computer Engineering, McGill University, Montreal, Canada, Tech. Rep., 2022.

[2] T. Lei, S. Chen, B. Wang, Z. Jiang, and N. Zou, "Adapted-MoE: Divide and Conquer for Anomaly Detection via Adapted Mixture of Experts," *arXiv preprint arXiv:2409.05611*, Sep. 2024.

[3] Y. Zhao, G. Zheng, S. Mukherjee, R. McCann, and A. Awadallah, "ADMoE: Anomaly Detection with Mixture-of-Experts from Noisy Labels," in *Proc. AAAI Conf. Artificial Intelligence*, vol. 37, 2023.

[4] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," *ACM Comput. Surveys*, vol. 41, no. 3, pp. 1–58, 2009.