Pluto: Sample Selection for Robust Anomaly Detection on Polluted Log Data

LEI MA, Worcester Polytechnic Institute, USA
LEI CAO, University of Arizona, USA
PETER M. VANNOSTRAND, Worcester Polytechnic Institute, USA
DENNIS M. HOFMANN, Worcester Polytechnic Institute, USA
YAO SU, Worcester Polytechnic Institute, USA
ELKE A. RUNDENSTEINER, Worcester Polytechnic Institute, USA

Log anomaly detection, critical in identifying system failures and preempting security breaches, finds irregular patterns within large volumes of log data. Modern log anomaly detectors rely on training deep learning models on clean anomaly-free log data. However, such clean log data requires expensive and tedious human labeling. In this paper, we thus propose a robust log anomaly detection framework, Pluto, that automatically selects a clean representative sample subset of the polluted log sequence data to train a Transformer-based anomaly detection model. Pluto features three innovations. First, due to localized concentrations of anomalies inherent in the embedding space of log data, Pluto partitions the sequence embedding space generated by the model into regions that then allow it to identify and discard regions that are highly polluted by our pollution level estimation scheme, based on our pollution quantification via Gaussian mixture modeling. Second, for the remaining more slightly polluted regions, we select samples that maximally purify the eigenvector spectrum, which can be transformed into the NP-hard facility location problem; allowing us to leverage its greedy solution with a $(1 - \frac{1}{e})$ approximation guarantee in optimality. Third, by iteratively alternating between the above subset selection, a model re-training on the latest subset, and a subset filtering using dynamic training artifacts generated by the latest model, the data selected is progressively refined. The final sample set is used to retrain the final anomaly detection model. Our experiments on four real-world log benchmark datasets demonstrate that by retaining 77.7% (BGL) to 96.6% (ThunderBird) of the normal sequences while effectively removing 90.3% (BGL) to 100.0% (ThunderBird, HDFS) of the anomalies, Pluto provides a significant absolute F-1 improvement up to 68.86% (2.16% → 71.02%) compared to the state-of-the-art sample selection methods. The implementation of this work is available at https://github.com/LeiMa0324/Pluto-SIGMOD25.

CCS Concepts: • **Security and privacy** → **Intrusion/anomaly detection and malware mitigation**; • **Information systems** → *Data cleaning*; • **Computing methodologies** → *Artificial intelligence*.

Additional Key Words and Phrases: Anomaly detection, Log sequence, Sample selection, Polluted data

ACM Reference Format:
Lei Ma, Lei Cao, Peter M. VanNostrand, Dennis M. Hofmann, Yao Su, and Elke A. Rundensteiner. 2024. Pluto: Sample Selection for Robust Anomaly Detection on Polluted Log Data. *Proc. ACM Manag. Data* 2, 4 (SIGMOD), Article 203 (September 2024), 25 pages. https://doi.org/10.1145/3677139

Authors' Contact Information: Lei Ma, lma5@wpi.edu, Worcester Polytechnic Institute, USA; Lei Cao, caolei@arizona.edu, University of Arizona, USA; Peter M. VanNostrand, pvannostrand@wpi.edu, Worcester Polytechnic Institute, USA; Dennis M. Hofmann, dmhofmann@wpi.edu, Worcester Polytechnic Institute, USA; Yao Su, ysu6@wpi.edu, Worcester Polytechnic Institute, USA; Elke A. Rundensteiner, rundenst@wpi.edu, Worcester Polytechnic Institute, USA.

ACM 2836-6573/2024/9-ART203
https://doi.org/10.1145/3677139

1 Introduction

Log anomaly detection, widely used to reveal intrusions, identify software bugs, or capture system failures [5], can be utilized in data systems including high-performance computing systems, data center networks, and IoT systems. To deal with the increasingly complex log data produced by these modern data systems, researchers have started to use deep learning models such as LSTM, RNN, and Transformers to detect log anomalies [8, 11, 16, 42, 51]. Intuitively, these works aim to learn the representation of normal log sequences and then detect abnormal log sequences via their deviation from the learned normal representation. However, these methods rely on the availability of a clean training set not polluted by anomalies (abnormal sequences), which, unfortunately, is extremely expensive to obtain. In addition, these methods have been shown to degrade dramatically even if the training data is only slightly polluted, due to the corrupted learned representation [20] – which we confirm later in our experimental study (Sec. 9.1).

To address this challenging problem, we propose Pluto to extract a clean and representative subset from the polluted dataset on the sequence level, which can then be used to train the anomaly detection model. Following most of the anomaly detection research [13], we select sequences mainly in the feature space, using sequence embeddings generated by a Transformer-based anomaly detection model. We motivate our sample selection task and showcase the challenges by bridging a gap between the root causes of the abnormal log sequences and their consequent characteristics in the embedding space.

Real-World Log Data Example. Figure 1 shows four example log sequences from a real log dataset HDFS [46] and a visualization of the embedding space of the whole dataset. On the left, we show two pairs of similar log sequences $S1$, $S2$, and $S3$, $S4$, where $S1$, $S3$, and $S4$ are abnormal, and $S2$ is normal. On the right, we show the visualization of the embedding space of the whole log sequence dataset, with a zoom-in view of two embedding regions $R1$ and $R2$ of the example sequences pairs.

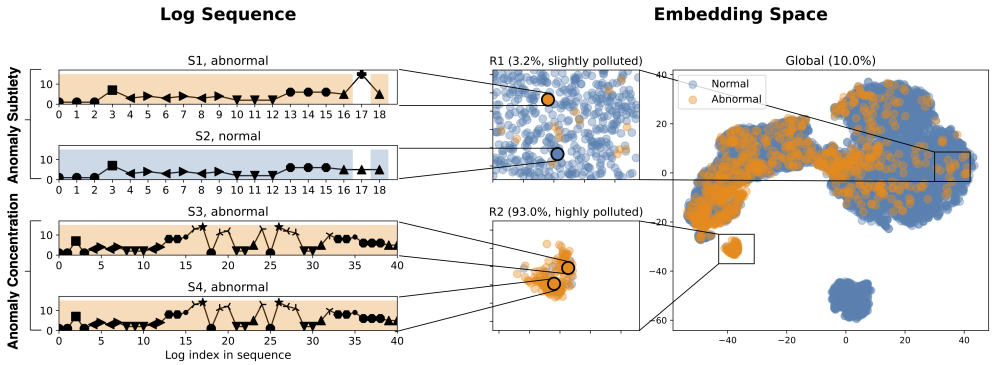


Fig. 1. Visualization of four example sequences and the embedding space of the HDFS log sequence dataset. Left: visualization of the log sequence, where the x-axis is the log index in the sequence, and the y-axis is the log key. Each unique log key is represented by a unique marker. The common subsequences between a pair are highlighted in the colored shade. Right: 2D t-SNE [41] visualization of the sequence embeddings of the HDFS dataset generated by Transformer after 200 epochs. Global anomaly ratio: 10.0%, region $R1$ anomaly ratio: 3.2%, region $R2$ anomaly ratio: 93.0%.

Challenges. We derive the following three challenges by observing the anomaly distribution in this dataset – emblematic of similar phenomena in other log data sets[1].

[1]See Section 9.2 for the full visualization of all four datasets: BGL, HDFS, ThunderBird, and Spirit.

Challenge 1: Anomaly subtlety with slight pollution. By comparing the normal sequence $S1$ and the abnormal one $S2$, we find that $S2$ is identical to $S1$ except for the log key at index 17, which indicates that an anomaly could be very subtle with a small deviation from the normality. Such subtlety of $S2$ as an anomaly makes it close to $S1$ in the corresponding embedding space shown in region $R1$. The gathering of normal sequences with a few subtle anomalies causes slight pollution in the local region, where anomalies are not necessarily far away from normal data as usually assumed. Thus, a sample selection method on the embedding space must be able to select normal sequences correctly when facing the interference of subtle anomalies in slightly polluted regions.

Challenge 2: Anomaly concentration with high pollution. When abnormal sequences can be similar to normal sequences like $S1$ and $S2$, they can also be similar or even identical to each other like $S3$ and $S4$. The identicalness between the abnormal sequences can be explained by the fixed exception paths in large systems. Some modules or nodes may be prone to certain failures, frequently generating failure-related logs in a fixed pattern, which eventually creates identical abnormal sequences. The concentration of these identical abnormal sequences brings high (or even pure) pollution in the local region, such as $R2$. Thus, to obtain a clean training set, the selection method has to be robust to the extreme anomaly concentration in highly polluted regions.

Challenge 3: Imbalanced pollution across embedding space. The co-existence of anomaly subtlety and anomaly concentration causes uneven pollution in the overall embedding space, with different areas of the embedding space experiencing diverse concentrations of pollution levels, as in $R1$ and $R2$. Our experiments (Sec. 9.3) show that state-of-the-art selection methods [12, 18, 37] are rendered insufficient since these pollution-agnostic selection approaches will either oversample anomalies or undersample normal data. Instead, the selection method must be sensitive and adapt to these uneven levels of pollution to obtain a representative yet clean training set.

State-of-the-Art. The unique challenges of log data will compromise the effectiveness of the state-of-the-art sample-selection methods for noisy data. We elaborate below and summarize in Table 1.

Category	Selection metric	Anomaly Subtlety	Anomaly Concentration	Imbalanced Pollution
Co-teaching	L	✓	–	–
ITLM	L	✓	–	–
FINE	R	✓	partial	–
TopoFilter	RP	✓	partial	–
Pluto (ours)	**RPL**	✓	✓	✓

Table 1. State-of-the-Art vs. challenges. (R: Representation. P: Predicted labels. L:Loss)

Most sample selection methods [6, 12, 17, 37, 47, 49] adopt the *Small-loss trick* heuristic, namely, they select samples with small loss as being potentially clean. This rests on the observation [2, 39] that a model first learns the core patterns in the data before overfitting to the more unusual and noisy parts of the data. Although the *Small-loss trick* has been shown to perform well on balanced data with synthetic random noise of a low rate, recent works [48, 52] point out that this may fail under a high noise ratio. In our case, with the existence of concentrated anomalies as in $R2$, the model risks quickly overfitting such anomaly patterns, compromising the effectiveness of this standard *Small-loss trick* approach.

Other recent works [18, 43] utilize properties of the latent space representation as the criterion to distinguish clean and noisy samples. Similar to loss-based methods, these techniques also assume a global yet universally low rate of random noise. Under these assumptions, they consider either the principal component of the data [18] or the largest connected graph in the latent space to be

clean [43] as guidance in selecting samples. Although they claim more robustness to a somewhat higher rate of random noise by leveraging the topological information other than the *Small-loss trick*, we demonstrate in our experiments in Sec. 9.3 that these methods don't effectively work when concentrated anomalies exist, as in R2, due to discrepancies between (1) real anomalies with patterns vs. assumed random noise and (2) real imbalanced global pollution vs. assumed universal global pollution.

Proposed Method. Given a polluted log sequence dataset, our proposed solution PLUTO automatically selects a clean and representative training set to train a Transformer-based anomaly detection model without any label supervision. Due to the imbalanced pollution inherent in log sequence data, PLUTO first divides the embedding space into regions via clustering to support region-specific pollution estimation. Based on our pollution quantification theory and dominance metric, PLUTO addresses the challenges of anomaly concentration and imbalanced pollution by accurately identifying and discarding the highly polluted regions, keeping the slightly polluted regions for the subsequent fine-grained sample selection. To address the anomaly subtlety problem when sampling from the remaining slightly polluted regions, PLUTO utilizes a *Spectrum-purifying* sample selection strategy to select samples (sequences) that maximally purify the eigenvector spectrum. By applying the selection strategy on all slightly polluted clusters, PLUTO generates a *base subset* based only on sequence embeddings.

Since the *base subset* is free from concentrated anomalies, PLUTO can now effectively leverage the traditional dynamic data artifacts, including sequence loss and predicted anomaly labels generated by the model, to filter out the model-recognized anomalies from the data subset. By iteratively alternating between the *base subset* selection, filtering, and model re-training, PLUTO refines the subset in a self-evolving fashion so that this subset can be used to retrain the anomaly detection model.

Contributions. Our work provides the following contributions.

1. We present PLUTO, an iterative framework to effectively select a clean representative subset from a polluted log dataset to train an anomaly detection model. PLUTO is the first sample-selection-driven solution for robust log anomaly detection.

2. We propose a novel strategy for quantifying pollution of a region in the embedding space by modeling it as a Gaussian mixture with the theoretical insights of *Component weights estimation* and *Spectroscopic estimation*. This facilitates accurate estimation of pollution level of a region, and identify the highly polluted ones.

3. We develop a *Spectrum-purifying* sample selection strategy for the slightly polluted regions, aiming to remove samples aligned with the minor eigenvectors. By transforming this problem into an NP-hard *Facility location* problem, we adopt the classic greedy solution with a suboptimal guarantee.

4. We design an iterative subset refinement process, which enhances the quality of the selected sample subset by leveraging dynamic data artifacts generated by the model on this cleaner data.

5. Our experimental studies on four real-world log datasets demonstrate that through sample selection, PLUTO keeps 77.7% (BGL) to 96.6% (ThunberBird) of the normal sequences, meanwhile removing 90.3% (BGL) to 100.0% (ThunderBird, HDFS) of the anomalies. This results in it achieving an absolute F-1 improvement of up to 68.86%.

2 Preliminaries

Below, we introduce log preprocessing and log anomaly detection. Then, we formally introduce our sample selection problem for robust log anomaly detection with polluted data. Please refer to Table 2 for notations used in this work.

Notation	Description
k	log key
S	A sequence of log keys
$\mathcal{D}$	A dataset of log key sequences
$\bar{\mathcal{D}}$	The selected subset from $\mathcal{D}$
r	Anomaly ratio of $\mathcal{D}$
$\boldsymbol{C_i}$	The i-th cluster in $\mathcal{D}$
$\bar{\boldsymbol{C}}_{\boldsymbol{i}}$	The selected subset of the cluster $\boldsymbol{C_i}$
r_i	Anomaly ratio of the cluster $\boldsymbol{C_i}$
$\boldsymbol{E_i}$	Embedding matrix of the cluster $\boldsymbol{C_i}$
$\boldsymbol{\mathcal{N}_+}$	Normal component of a Gaussian mixture
$\boldsymbol{\mathcal{N}_-}$	Abnormal component of a Gaussian mixture

Table 2. Notations.

2.1 Log Pre-processing

Log Parsing. Log parsing methods [7, 14] focus on parsing text-based log events into structured data with defined columns, including timestamps, log level, source, etc. Given a file with different logs, a log parser analyzes the similarity between log text and generates the log templates with parameters indicated as <*>. After parsing, each log will be assigned a log template ID (or, log key ID) with its own parameters, as well as the attributes of the defined columns.

Log Partitioning. After parsing, the structured logs are partitioned into log sequences for pattern learning. The most common methods partition logs via identifiers or time-based windows [16]. As Log files from real-world high-computing systems usually have interleaving logs from different modules/nodes, partitioning the logs by a certain identifier like node ID, domain ID, or file block ID makes the execution patterns of each node/module easier to identify. On the other hand, time-based partitioning slices long log files into either overlapping or non-overlapping small sequences, which will be fed into the downstream tasks for log pattern learning. These two partitioning methods can be used either individually or collaboratively depending on the use case.

2.2 Sample Selection Problem for Robust Log Anomaly Detection

Definition 1 (Log). A log message (or, in short, log) l is generated by code as a formatted string $l = [tok_1, tok_2, \ldots, tok_{|l|}]$, using a log template T with a parameter list P. Specifically, $|l|$ denotes the total number of tokens (words) in the string, and tok_i represents the i-th token, which can be either a word, symbol, or value.

A log message l can be assigned with an anomaly label $y \in \{0, 1\}$ as being normal or abnormal by an expert, by simply examining its content, such as an error message, or by its context, such as abnormal co-existence with other logs [19].

Definition 2 (Log key). A log key k is considered a unique identifier for a log template T. After parsing, each log l is mapped to a log key k based on its template T.

Similar to the most log anomaly detection works [8, 11, 42], we consider a log key k share the same anomaly label y with its log l, *i.e.*, , the log key k is labeled as abnormal when its log l is abnormal.

Definition 3 (Log key sequence). A log key sequence $S = [k_1, k_2, \ldots, k_{|S|}]$ is a sequence consisting of log keys in time order under an observed time window, where k_i represents the i-th log key in the sequence and $|S|$ represents the sequence length.

Given a log key sequence $S = [k_i]_{i=1}^{|S|}$ with the anomaly labels $[y_i]_{i=1}^{|S|}$ of all its log keys k_i, we adopt the common definition of the *abnormal log key sequence* below used by most log anomaly detection works [8, 11, 42].

Definition 4. [**Abnormal log key sequence**] Given a log key sequence $S = [k_1, k_2, \ldots, k_{|S|}]$ with the anomaly labels of the log keys $Y = [y_1, y_2, \ldots, y_{|S|}]$, the anomaly label y^* of S is defined as $y^* = \bigwedge_{i=1}^{n} y_i$, *i.e.*, S is abnormal if it contains at least abnormal log keys; otherwise it is normal.

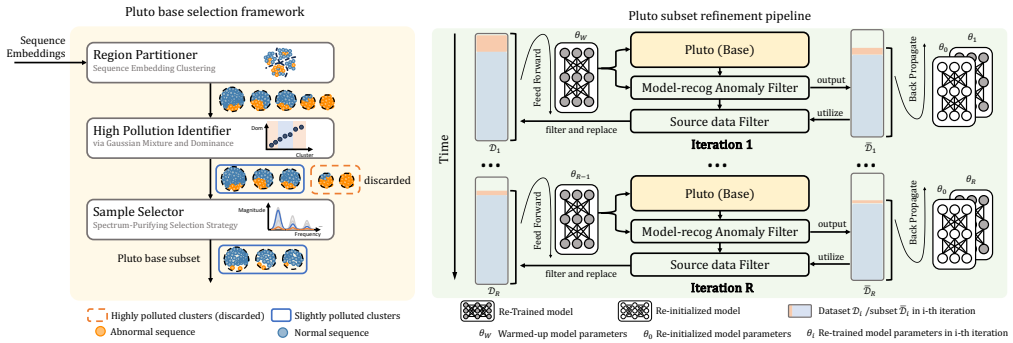
Definition 5 (Log key sequence anomaly detection). Given a log key sequence dataset $\mathcal{D} = \{S_i\}_{i=1}^{|\mathcal{D}|}$ of only normal sequences, an anomaly detection model $f_\theta(\cdot)$ learns the normal representation of $\mathcal{D}$, so that during the inference time, given a new sequence S, it detects whether S is abnormal or not by measuring its deviation from the learned representation.

We now define our sample selection problem for robust log key sequence anomaly detection with polluted datasets, meaning, when the training set contains both normal and abnormal sequences.

Definition 6 (Sample selection for robust log key sequence anomaly detection). Given a polluted unlabeled log key sequence dataset $\mathcal{D} = \{S_i\}_{i=1}^{|\mathcal{D}|}$ with anomaly ratio r, and an anomaly detection model $f_\theta(\cdot)$, the problem of **sample selection for robust log key sequence anomaly detection** is to select a representative subset $\bar{\mathcal{D}} \subset \mathcal{D}$ with $\bar{\mathcal{D}}$ cleaner than $\mathcal{D}$, with anomaly ratio $\bar{r} \ll r$, so that $f_\theta(\cdot)$ trained on $\bar{\mathcal{D}}$ achieves a better anomaly detection performance than trained on the original polluted dataset $\mathcal{D}$.

3 Pluto Overview

Figure 2 depicts the Pluto overview consisting of two components: base selection (Figure 2(a)) and subset refinement (Figure 2(b)). As the core component, the Pluto base selection aims to address the main challenges of anomaly subtlety, anomaly concentration and imbalanced pollution, and select a *base subset*. Then the Pluto subset refinement further refines the *base subset* iteratively so that the *refined subset* in the last iteration can be used as the training set to retrain the anomaly detection model.



(a) Base selection (b) Subset refinement

Fig. 2. Pluto overview. a): the static Pluto base selection framework using the given sequence embeddings. b): the Pluto iterative subset refinement pipeline with two example iterations, with Pluto base selection from (a) as one component in each iteration.

Base Selection. As in Figure 2(a), Pluto utilizes three modules in the base selection component: the region partitioner, pollution level estimator (Sec. 4, 5), and sample selector (Sec. 6). Given the sequence embeddings of the polluted dataset, due to the imbalanced pollution in different regions, the region partitioner first divides the embedding space into regions via a clustering algorithm[2].

[2]The words "cluster" and "region" are henceforth used interchangeably in this paper

Then, by modeling a region (cluster) as a Gaussian mixture with our pollution quantification theory (Sec. 4), the high pollution identifier (Sec. 5) uses the dominance metric to identify and discard highly polluted clusters. This addresses the anomaly concentration problem.

Then, to solve the anomaly subtlety problem for the remaining slightly polluted clusters, the sample selector adopts a *Spectrum-purifying* selection strategy (Sec. 6) which best removes the anomalies. By transforming the selection problem into a *Facility location* problem, our sample selector achieves a clean selection by adopting the classic greedy algorithm which comes with approximation bound guarantees. By selecting samples from each slightly polluted cluster, Pluto obtains the *base subset* that is free from issues of high anomaly concentration and imbalanced pollution.

Subset Refinement. Although the *base subset* is free of anomaly concentration, it may still contain anomalies that cannot be identified simply by the sequence embeddings. Thus, we incorporate the model's dynamic perspective by leveraging its generated data artifacts to further refine the *base subset* in an iterative manner. Figure 2(b) shows the first and the last (R-th) iterations of subset refinement (Sec.7).

In the i-th iteration, Pluto feeds the polluted dataset (source data) $\mathcal{D}_i$ into the Transformer-based anomaly detection model $f_{\theta_{i-1}}(\cdot)$ (warmed-up model $f_{\theta_W}(\cdot)$ for the first iteration) trained from the last iteration, to obtain the sequence embeddings, loss, and predicted anomaly labels. Then, the above Pluto base selector takes the sequence embeddings to generate the *base subset*. With the *base subset*, loss, and the predicted labels, the model-recognized anomaly filter filters out potential anomalies recognized by the model from the *base subset* to obtain subset $\bar{\mathcal{D}}_i$. Using the filtered subset $\bar{\mathcal{D}}_i$ as the new training set, Pluto re-trains the model for G epochs so that the current model $f_{\theta_i}(\cdot)$ is upgraded from the later iteration $f_{\theta_{i-1}}(\cdot)$ due to the cleaner training set. After the selection, a source data filter is leveraged to use the selected subset $\bar{\mathcal{D}}_i$ to, in turn, clean the original source data $\mathcal{D}_i$ to obtain the source data for the next iteration $\mathcal{D}_{i+1}$, so that the selected subset $\bar{\mathcal{D}}_{i+1}$ in the next iteration is cleaner than the currently selected subset $\bar{\mathcal{D}}_i$ due to a cleaner selection source. After R iterations of refinement, Pluto succeeds to clean both the source data and the subset simultaneously and progressively, so that the last and cleanest subset $\bar{\mathcal{D}}_R$ can be used to re-train the anomaly detection model for deployment.

4 Pollution Quantification via Gaussian Mixture

Given the sequence embeddings of the dataset $\mathcal{D}$, the region partitioner first applies clustering, such as K-means, to obtain m clusters of embeddings $\{\boldsymbol{E_i}\}_{i=1}^{m}$ for the corresponding clusters of sequences $\{\boldsymbol{C_i}\}_{i=1}^{m}$. To correctly select normal samples in different clusters, here we provide a theoretically grounded solution to quantify the pollution in a cluster of embeddings $\boldsymbol{E_i}$ by modeling it as a Gaussian mixture of normal and abnormal components with a two-step pollution approximation. For simplicity, we drop the subscript of $\boldsymbol{E}$ and $\boldsymbol{C}$ in the rest of this section.

Following existing linear discriminant analysis (LDA) [9, 18, 21] and anomaly detection surveys [34], we model the embedding matrix $\boldsymbol{E}$ of a cluster as a Gaussian mixture $\mathcal{N}$ of two Gaussian components $\mathcal{N}_+$ and $\mathcal{N}_-$ corresponding to the normal and abnormal data in the cluster, with respective weights w_+ and w_- ($w_+ + w_- = 1$) as follows:

$$\mathcal{N} = w_+\mathcal{N}_+(\mu_+,\ \sigma_+^2) + w_-\mathcal{N}_-(\mu_-,\ \sigma_-^2) \tag{1}$$

Since the weights of the components represent their corresponding portions, the ratio of the weights $\frac{w_-}{w_+}$ thus effectively indicates how heavily this cluster is polluted. However, without being aware of the actual $\mathcal{N}_+$ and $\mathcal{N}_-$ distributions, it seems impossible to compute the ratio $\frac{w_-}{w_+}$, while it is non-trivial to learn the two distributions without any prior knowledge. To solve this problem,

we propose to design a strategy to directly approximate the ratio $\frac{w_-}{w_+}$, without having to model the two individual Gaussians beforehand.

Together, our theoretical insights and empirical observations, namely, *Component weights estimation* and *Spectroscopic estimation*, allow us to build this effective approximation solution.

Component Weights Estimation. Let $\boldsymbol{v}_+$ and $\boldsymbol{v}_-$ be the centers of components $\mathcal{N}_+$ and $\mathcal{N}_-$ respectively. Then, we can write the mixture matrix $\boldsymbol{E}$ in the following spectral form with $\boldsymbol{v}_+$ and $\boldsymbol{v}_-$:

$$\boldsymbol{E} \approx \lambda_+ \boldsymbol{u_+} \cdot \boldsymbol{v_+} + \lambda_- \boldsymbol{u_-} \cdot \boldsymbol{v_-} \tag{2}$$

where λ_+ and λ_- are scalars with the amplitude information and $\boldsymbol{u_+}$ and $\boldsymbol{u_-}$ are orthogonal unit vectors. Thus, the ratio of the components' weights $\frac{w_-}{w_+}$ in the mixture matrix $\boldsymbol{E}$ can be approximated by the *amplitude* ratio of the centers $\boldsymbol{v}_-$ and $\boldsymbol{v}_+$ in the mixture matrix $\boldsymbol{E}$, corresponding to $\frac{\lambda_-}{\lambda_+}$. However, Equation 2 is non-trivial to derive without knowing the actual centers $\boldsymbol{v}_-$ and $\boldsymbol{v}_+$.

Spectroscopic Estimation. Following existing works [18, 26], we can consider the respective first singular vectors of the components $\mathcal{N}_+$ and $\mathcal{N}_-$ as their centers $\boldsymbol{v_+}$ and $\boldsymbol{v_-}$. In this case, the theory of *Spectroscopic estimation* [38] enables us to empirically approximate the centers $\boldsymbol{v}_+$ and $\boldsymbol{v}_-$ via the mixture matrix $\boldsymbol{E}$ without knowing $\mathcal{N}_+$ and $\mathcal{N}_-$. That is, the top two singular vectors of the mixture matrix are nearly identical to the first singular vectors of the two individual Gaussian components. Thus, with the following rank-2 approximation of $\boldsymbol{E}$ after Singular Value Decomposition (SVD):

$$\begin{aligned} \boldsymbol{E} &= U\Sigma V^T \\ &\approx \lambda_1 \boldsymbol{u_1} \cdot \boldsymbol{v_1} + \lambda_2 \boldsymbol{u_2} \cdot \boldsymbol{v_2} \end{aligned} \tag{3}$$

we approximate $\boldsymbol{v_+}$ and $\boldsymbol{v_-}$ in Equation 2 by $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ in Equation 3, and the empirical top two singular values λ_2 and $\lambda_1 (\lambda_1 > \lambda_2)$ of $\boldsymbol{E}$ can be used to approximate $\frac{\lambda_-}{\lambda_+}$, thus the ratio $\frac{w_-}{w_+}$.

With the above approximation of $\frac{w_-}{w_+}$, one may conclude that a larger $\frac{\lambda_2}{\lambda_1}$ indicates higher pollution. However, this only holds under the assumption of a larger normal component that $w_+ > w_-$, so that the top singular vector $\boldsymbol{v_1}$ and singular value λ_1 correspond to the normal component, which is not always true in all of our scenarios, for example, region *R2* in Figure 1. Next, we show that the estimation of the ratio $\frac{w_-}{w_+}$ varies depending on the different dominating relationships between the components $\mathcal{N}_+$ and $\mathcal{N}_-$.

5 High Pollution Identification

Based on the above pollution quantification theory, we now describe how we identify the clusters with relatively high pollution levels, which then will be discarded. While the remaining slightly polluted clusters are kept for the downstream sample selection task (Sec. 6).

Based on the pollution quantification and our dominance metric, we estimate the coarse-grained pollution level of a cluster based on its dominating relationship between the normal and abnormal components. Specifically, we consider the slightly polluted clusters to be absolutely dominated by the normal component, following the common anomaly rarity setting in most anomaly detection research [19]. Thus, apart from the slightly polluted clusters, we aim to identify two cases of highly polluted clusters, extreme and mild, where the abnormal component is either absolutely dominating in the former or balanced with the normal component in the latter. These different component dominating relationships of the two cases require a distinct treatment during identification.

First, we define the dominance metric.

Definition 7 (*Cluster dominance*). Given the embedding matrix $\boldsymbol{E}$ of a cluster, let λ_1 and $\lambda_2 (\lambda_1 > \lambda_2)$ be the first and second singular values of $\boldsymbol{E}$, the dominance of the cluster is defined as $dom = \lambda_1 / \lambda_2$.

According to the *Pollution quantification* (Sec. 4), the above defined dominance metric represents the relative size of the larger component compared to the smaller one. By using the dominance and examining the different dominating relationships, we conclude that the pollution is inversely proportional to the dominance of clusters with slight or mildly high pollution and is proportional to the dominance of clusters with extremely high pollution, where this conclusion can be leveraged to decide the clusters' coarse-grained pollution level. We elaborate as follows.

Extremely High Pollution. The clusters with large dominance could have extremely high pollution when the abnormal component has a much larger weight $w_- \gg w_+$, according to Equation 1. In this case, the first singular vector and value $\boldsymbol{v_1}$ and λ_1 corresponds to the abnormal component so that the ratio $\frac{w_-}{w_+}$ is proportional to the dominance as follows:

$$\frac{w_-}{w_+} \propto \frac{\lambda_1}{\lambda_2} = dom \tag{4}$$

However, these extremely highly polluted clusters can be hard to distinguish from slightly polluted ones without knowing the ground truth dominant component. We solve this problem by finding that a cluster with extremely high pollution tends to have an abnormally higher dominance than other clusters. In contrast, slightly polluted clusters tend to have similarly high dominance, based on the reason of anomaly concentration in log data. Since such an extremely highly polluted cluster contains very few normal sequences, the abnormal sequences in it are globally different from any normal sequences but similar to each other, for example, region $R2$ in Figure 1 (Sec. 1). Thus, instead of being a deviation from the normal sequences, these abnormal sequences have a consistent and unique pattern. This is likely caused by a specific error loop that is not part of the normal execution. The abnormal sequences generated in this scenario can be highly identical; so that the embedding matrix $\boldsymbol{E}$ of the cluster has $rank(\boldsymbol{E}) = O(1)$ with a much steeper exponential decay in singular values, eventually causing a spike in dominance compared to other clusters. We support this conclusion by the following observations on the real log data.

Observation 1 (Dominance spike of clusters with high pollution (Extreme)). Figure 3 shows the anomaly ratios and dominance of 20 clusters of the sequence embeddings produced from the ThunderBird dataset. The clusters are sorted by dominance in ascending order. Clusters 0-18 are clean, while Cluster 19 contains only anomalies. As we can see, the dominance of clean clusters increases gradually, while it suddenly rises on cluster 19 and causes an elbow on the curve.

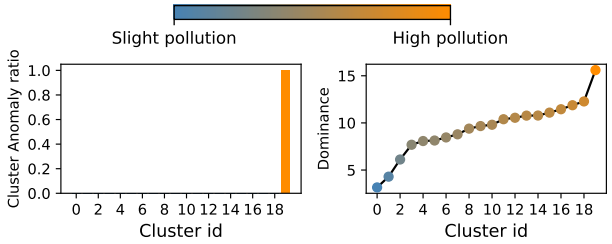


Fig. 3. Cluster anomaly ratio and dominance (ThunderBird dataset), sorted by dominance in ascending order.

Observation 2 (Fast singular value decay of clusters with high pollution (Extreme)). Figure 4 shows the ratio between the i-th singular value λ_i and the first singular value λ_1 for clusters 18 and 19 with the highest dominance values. Due to the high similarity between the anomalies inside cluster 19, its singular values decrease faster than those of the clean cluster 18. This caused its abnormally high dominance in Figure 3.

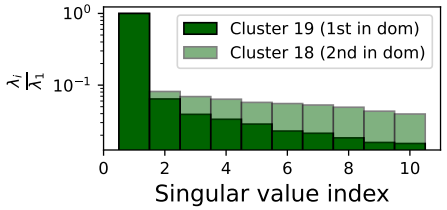


Fig. 4. Singular value decay of clusters 18 and 19 (Thunderbird).

Based on this insight, we consider a cluster highly polluted (Extreme) if its dominance causes the greatest slope change (detected by a knee-point detector [35]) as well as ranked in the top p percent clusters with maximal dominance.

Mildly High Pollution. Unlike slightly and extremely highly polluted clusters with large dominance for the absolute dominating component (normal or abnormal), the mildly highly polluted clusters will have a smaller dominance since a small dominance means that λ_1 is close to λ_2, indicating more balanced components, no matter which one is larger. Based on this insight, given a percent threshold p, we consider the top p percent clusters with minimal dominance to have high pollution (Mild).

By identifying the above highly polluted clusters, the remaining clusters can be safely considered as slightly polluted.

To better facilitate the sample selection, given the global anomaly ratio, we also estimate the anomaly ratio of each cluster.

Cluster Anomaly Ratio Estimator. According to the above analysis, the pollution is inversely proportional to the dominance of clusters with slight or mildly high pollution and is proportional to the dominance of clusters with extremely high pollution.

Thus, we estimate the anomaly ratio r_i of a cluster $\boldsymbol{C_i}$ with a coefficient α as follows:

$$\frac{r_i}{1-r_i} = \begin{cases} \frac{w_-}{w_+} = \frac{1}{dom_i} \times \alpha & \boldsymbol{C_i} \text{ is slight or mild} \\ \frac{w_-}{w_+} = dom_i \times \alpha & \boldsymbol{C_i} \text{ is extreme} \end{cases} \tag{5}$$

Let r be the anomaly ratio of the whole dataset $\mathcal{D}$, and m be the number of clusters, the number of anomalies in all clusters should sum up to the total number of anomalies as follows:

$$\sum_{i=1}^{m} r_i \times |C_i| = r \times |\mathcal{D}| \tag{6}$$

Given the m clusters with their estimated pollution levels, we plug the corresponding cluster anomaly ratio estimator from Equation 5 into Equation 6 to compute α. We then plug α back into Equation 5 to compute the estimated anomaly ratio of each cluster. After discarding the highly polluted clusters, the remaining slightly polluted clusters along with their estimated anomaly ratios are kept. Due to limited space, we provide the HighPollutionIdentify algorithm in the Supplement[3]. Our experimental results in Section 9.3 show that Pluto accurately detects the highly polluted clusters in real-world data sets, which contain up to 98.4% of all anomalies.

6 Spectrum-Purifying Selection

With the highly polluted clusters discarded, we can now safely apply sample selection to the remaining (slightly polluted) clusters using the assumption that all clusters are dominated by normal sequences. Using the estimated anomaly ratios computed above, we now design a spectrum-purifying sample selection strategy.

The key to successful sample selection is to find a reliable criterion to distinguish between normal and abnormal samples. Since now most sequences are normal, the first eigenvector $\boldsymbol{v_1}$ of

the embedding matrix $\boldsymbol{E}$ of a slightly polluted cluster is close to its ground-truth normal center $\boldsymbol{v_+}$. Thus, a natural idea may be to use the first eigenvector $\boldsymbol{v_1}$ as the criterion to select sequences that are close to it [18]. We refer to this as *Spectrum-picking* strategy. However, although normal sequences are indeed aligned with the first eigenvector, we observe that the perturbation from the anomalies may compromise the first eigenvector's ability to distinguish normal and abnormal sequences.

Observation 3 (Sequences' differential alignment to eigenvectors.). Figure 5 shows the anomaly ratio of a real cluster from the BGL dataset and its sequences' cosine similarity to its top three eigenvectors, respectively. While the normal and abnormal sequences are distributed overlappingly in the cosine similarity to the first eigenvector, they are more separable by cosine similarity to the second and third eigenvectors - that is, abnormal sequences have larger relative cosine similarity.

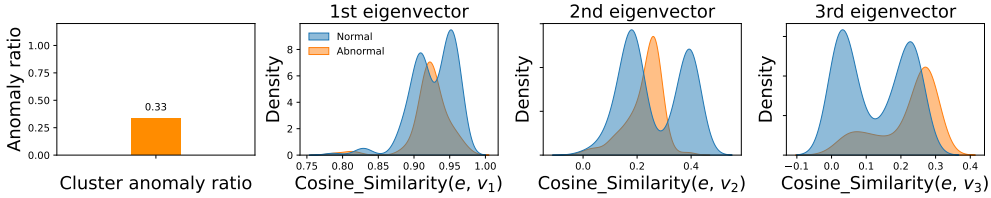


Fig. 5. Sequence cosine similarity to the top 3 eigenvectors in a real cluster (BGL dataset, better viewed in color) .

Such observation can be explained by the eigenvectors' different levels of sensitivity to perturbation of anomalies. Figure 6 illustrates the top three eigenvectors before and after the perturbation from the anomalies, where the first eigenvector is more robust than the minor ones. As the major factor of the slightly polluted cluster, the first eigenvector represents the majority which should be more robust to perturbation; meanwhile, the minor eigenvectors, as smaller factors, are more vulnerable.

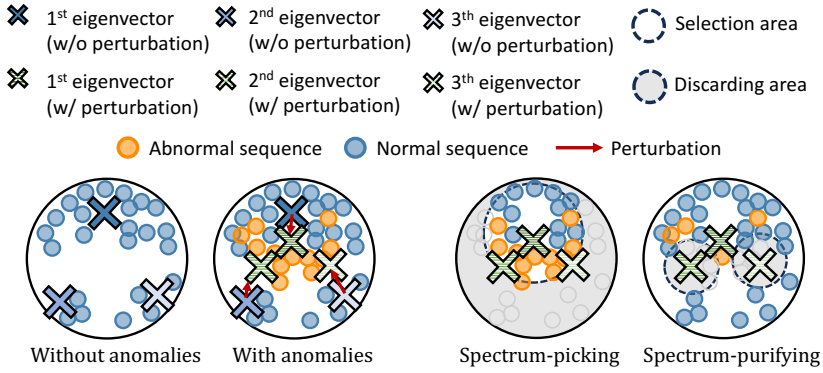


Fig. 6. Eigenvectors' sensitivity to anomaly perturbation.

Fig. 7. Different selection strategies.

With a considerable number of anomalies, when the perturbed first eigenvector starts to fall between normal and abnormal sequences, losing its identifiability for normal sequences, the minor

eigenvectors will be relatively more perturbed and closer to anomalies. They thus can be used to identify the anomalies. This conclusion is consistent with the well-known fact that an eigenvector will exhibit more sensitivity to perturbation when its corresponding eigenvalue lies near another eigenvalue [27] .

Thus, instead of applying the *Spectrum-picking* strategy using the first eigenvector, we adopt a *Spectrum-purifying* strategy leveraging the minor eigenvectors, with the aim to remove anomalies aligned with them. Figure 7 shows the difference between these two selection strategies. Instead of an exhaustive search and sample removal on the whole spectrum, we achieve this goal by finding k-medoids of the cluster. This has been shown to effectively discard samples associated with minor eigenvectors [30]. By assuming a matrix can be divided into a low-rank *information space* which contains the important information, and a *nuisance space* with noise [30, 33], the k-medoids problem is capable of finding k most representative samples in the low-rank *information space* and discard the samples scattered in the *nuisance space* associated with the minor eigenvectors. Formally, given a cluster of sequence embeddings $\boldsymbol{E}$, it selects the optimal subset $\bar{\boldsymbol{E}}^*$ of size k out of all possible subsets as follows:

$$\bar{\boldsymbol{E}}^* = \arg\min_{\bar{\boldsymbol{E}} \subset \boldsymbol{E}} \sum_{\boldsymbol{e_i} \in \boldsymbol{E}} \min_{\boldsymbol{e_j} \in \bar{\boldsymbol{E}}} dist(\boldsymbol{e_i}, \boldsymbol{e_j}) \tag{7}$$

where $dist(\boldsymbol{e_i}, \boldsymbol{e_j})$ represents the distance between embedding $\boldsymbol{e_i}$ and $\boldsymbol{e_j}$.

By introducing a constant $d_0 > dist(\boldsymbol{e_i}, \boldsymbol{e_j})$ for any $\boldsymbol{e_i}, \boldsymbol{e_j} \in \boldsymbol{E}$, minimizing our problem in Equation 7 is equivalent to maximizing the following monotone submodular *Facility location* problem:

$$\bar{\boldsymbol{E}}^* = \arg\max_{\bar{\boldsymbol{E}} \subset \boldsymbol{E}} F(\bar{\boldsymbol{E}}) \tag{8}$$

$$F(\bar{\boldsymbol{E}}) = \sum_{\boldsymbol{e_i} \in \boldsymbol{E}} \max_{\boldsymbol{e_j} \in \bar{\boldsymbol{E}}} d_0 - dist(\boldsymbol{e_i}, \boldsymbol{e_j}) \tag{9}$$

Formally, a set function $F : 2^V \rightarrow \mathbb{R}$ is submodular if $F(X \cup \{x\}) - F(X) \geq F(Y \cup \{x\}) - F(Y)$ for every $X \subseteq Y \subseteq V$ and $x \in X \setminus Y$. F is monotone if $F(X) \leq F(Y)$ for every $X \subseteq Y$. Although this problem is NP-hard, we note that the classic greedy algorithm for maximizing the monotone submodular problem has been shown to provide a $(1 - \frac{1}{e})$-approximation [31]. In each step, the algorithm selects the sequence with maximal marginal utility $F(\bar{\boldsymbol{E}} \cup \{\boldsymbol{e}\}) - F(\bar{\boldsymbol{E}})$, with a complexity of $O(|\boldsymbol{E}|k)$, which can be further improved to $O(|\boldsymbol{E}|)$ by lazy evaluation [28, 29].

Thus, by leveraging the above greedy algorithm, given a slightly polluted cluster with its estimated anomaly ratio r_i, we purify the spectrum associated with minor eigenvectors by selecting $k = r_i|\boldsymbol{E}|$ medoids as the selected subset. Although finding k-medoids significantly removes samples aligned with minor eigenvectors, it is still possible that it contains representative anomalies when the anomalies form a strong pattern as the abnormal component represented by the second eigenvector. Thus, to minimize the selection error, after the general spectrum purification, we apply a special purification on the second eigenvector $\boldsymbol{v_2}$ by removing the $r_i|\boldsymbol{E}|$ sequences closer to $\boldsymbol{v_2}$.

BaseSelection algorithm. Given the required inputs, Algorithm 1 returns a *base subset* $\bar{\mathcal{D}}$ from the source dataset $\mathcal{D}$. It leverages two utility algorithms HighPollutionIdentify and GreedyFacilityLocation, which we include in the supplement[3] due to limited space. Specifically, Lines 2-7 apply the SVD on the embedding matrix of each cluster and calculate the respective dominance and top two eigenvectors. With the clusters' dominance, given a threshold p, Line 8 applies the utility algorithm HighPollutionIdentify to identify the sets of slightly and highly polluted clusters, $\boldsymbol{S}$ and $\boldsymbol{H}$ respectively, and estimate their anomaly ratios. Then, Lines 9-13 apply the *Spectrum-purifying* selection strategy on each slightly polluted cluster to obtain its selected subset $\bar{\boldsymbol{C}}_{\boldsymbol{i}}$ by leveraging the utility algorithm GreedyFacilityLocation and a special purification

Algorithm 1 BaseSelection

Input: Source dataset $\mathcal{D}$, Anomaly ratio r of $\mathcal{D}$, Threshold p, m Clusters $\{\boldsymbol{C_i}\}_{i=1}^{m}$, embedding matrices $\{\boldsymbol{E_i}\}_{i=1}^{m}$ of $\{\boldsymbol{C_i}\}_{i=1}^{m}$
Output: The base subset $\bar{\mathcal{D}}$

1: $\bar{\mathcal{D}} \leftarrow \emptyset$
▷ **Step 1: SVD for each cluster**
2: **for each** $\boldsymbol{E_i} \in \{\boldsymbol{E_i}\}_{i=1}^{m}$ **do**
3: $\boldsymbol{U}, \Sigma, \boldsymbol{V} = \text{SVD}(\boldsymbol{E_i})$
4: $\{dom_i\} \leftarrow \{dom_i\} \cup \frac{\text{The First Eigenvalue In } \Sigma}{\text{The Second Eigenvalue In } \Sigma}$
5: $\{\boldsymbol{v}_{1,i}\} \leftarrow \{\boldsymbol{v}_{1,i}\} \cup$ The First Column Of $\boldsymbol{V}$
6: $\{\boldsymbol{v}_{2,i}\} \leftarrow \{\boldsymbol{v}_{2,i}\} \cup$ The Second Column Of $\boldsymbol{V}$
7: **end for**
▷ **Step 2: High pollution identification**
8: $\boldsymbol{H}, \boldsymbol{S}, \{r_i\}_{i=1}^{m} \leftarrow$ HighPollutionIdentify$(\{dom_i\}_{i=1}^{m}, p)$
▷ **Step 3: Spectrum-purifying sample selection strategy**
9: **for each** $\boldsymbol{C_i} \in \boldsymbol{S}$ **do**
10: $\bar{\boldsymbol{C}}_{\boldsymbol{i}} \leftarrow$ GreedyFacilityLocation$(\boldsymbol{E_i}, \boldsymbol{C_i}, r_i)$
11: $\bar{\boldsymbol{C}}_{\boldsymbol{i}} \leftarrow \bar{\boldsymbol{C}}_{\boldsymbol{i}} -$ FindMaxCosSimSamples$(\boldsymbol{E_i}, \boldsymbol{C_i}, \boldsymbol{v}_{2,i}, r_i)$
12: $\bar{\mathcal{D}} \leftarrow \bar{\mathcal{D}} \cup \bar{\boldsymbol{C}}_{\boldsymbol{i}}$
13: **end for**
14: **return** $\bar{\mathcal{D}}$

on the second eigenvector. By unioning the selected subsets for all target clusters, Algo 1 returns the final *base subset* $\bar{\mathcal{D}}$ in Line 14.

Complexity Analysis. Let $|\mathcal{D}|$ be the total number of sequences, d be the embedding dimension, and m be the number of clusters, the complexity of Algo 1 is dominated by the complexity of SVD $O(m|\mathcal{D}|^2 d + md^3)$ [10], and the complexity of GreedyFacilityLocation algorithm $O(m|\mathcal{D}|d^2)$ [28, 29]. Due to the limited space, we provide the detailed complexity analysis in the Supplement[3].

7 Iterative Subset Refinement

Here, we propose our subset refinement that iteratively refines the *base subset* selected by the selection strategy. Our goal is to refine the selected subset iteratively so that the selected subset $\bar{\mathcal{D}}_i$ in the current iteration is cleaner than the selected subset $\bar{\mathcal{D}}_{i-1}$ in the last iteration. To this end, after the base selection in each iteration, we apply three additional steps to achieve refinement, namely, including *Model-recognized anomaly filtering*, *Model upgrading*, and *Souce data filtering*.

Model-recognized Anomaly Filtering. Existing sample selection methods [6, 12, 37, 49] use data artifacts generated by the model, such as loss, as the main criterion to identify the clean samples that are observed to have a small loss. However, due to their assumption of a low rate of random noise, our empirical evaluation (Sec. 9) shows that these methods cannot effectively perform as the main selection criteria with the existence of highly polluted clusters and concentrated anomalies since the anomalies with a consistent pattern can also have a small loss. However, since the concentrated anomalies are mostly removed from the *base subset*, Pluto now can effectively use the data artifacts as auxiliary selection criteria to further clean the subset. We thus leverage the sequence loss and predicted anomaly labels generated by the model to filter out the model-recognized anomalies from the *base subset* to get the *filtered subset*. Specifically, given a cluster with its estimated anomaly ratio r_i, we remove the top r_i ratio of sequences with the largest loss normed by the sequence length. Also, we maintain a set $\boldsymbol{A}$ that stores the historical predicted anomalies by the model during

iterations. These then will also be removed from the selected subset. The *filtered subset* in the last iteration is returned as the final *refined subset* for the whole refinement process.

Model Upgrading. After applying the *Model-recognized anomaly filtering*, we use the *filtered subset* to re-train the model for G epochs until the next selection iteration. With the subset being refined by iterations, the model learns a cleaner representation, which in turn progressively enhances the power of sequence loss and predicted anomalies as auxiliary selection criteria in the next iteration.

Source Data Filtering. To achieve progressive refinement, at the end of the $(i-1)$-th iteration after anomaly filtering and model upgrading, we clean the current source data $\mathcal{D}_{i-1}$ to obtain $\mathcal{D}_i$ for the next iteration, which will lead to a cleaner selected subset $\bar{\mathcal{D}}_i$ in the future than the currently selected subset $\bar{\mathcal{D}}_{i-1}$, due to a cleaner selection source. Specifically, given a source data $\mathcal{D}_{i-1}$, selecting a cleaner *filtered subset* $\bar{\mathcal{D}}_{i-1}$ from $\mathcal{D}_{i-1}$ must leave the unselected subset $\mathcal{D}_{i-1} - \bar{\mathcal{D}}_{i-1}$ more polluted than $\mathcal{D}_{i-1}$. Thus, we consider the intersection of the unselected subset $(\mathcal{D}_{i-1} - \bar{\mathcal{D}}_{i-1})$ and the historically predicted anomalies $\boldsymbol{A}$, as the consensus anomalies. It will then be removed from the source data $\mathcal{D}_{i-1}$ to generate $\mathcal{D}_i = \mathcal{D}_{i-1} - (\mathcal{D}_{i-1} - \bar{\mathcal{D}}_{i-1}) \cap \boldsymbol{A}$. With the cleaned source data $\mathcal{D}_i$, the subset selected $\bar{\mathcal{D}}_i$ in the i-th iteration will be further refined.

By incorporating comprehensive selection criteria and cleaning progressively, our experiment results (Sec. 9) show that Pluto subset refinement further significantly enhances the cleanliness of the *base subset*. Due to the limited space, we provide the SubsetRefine algorithm in a Supplement[3].

8 Experimental Setup

8.1 Datasets

We conduct our experimental study using four real-world log datasets generated by large high-performance computing systems. Three of them (BGL, HDFS, ThunderBird) are widely used by state-of-the-art log anomaly detection methods [8, 11, 16, 42, 51]. To give a more thorough evaluation, we introduce the fourth dataset, Spirit, which was collected together with ThunderBird but has not been used as commonly in the literature.

• **BlueGene/L** [32] (BGL) is a log data set collected from BlueGene/L supercomputer system at Lawrence Livermore National Labs (LLNL) in Livermore, California. Each log event contains an alert label, a timestamp, and the log content. This dataset contains 4,747,936 log messages, of which 348,460 are labeled as an alert.

• **Hadoop Distributed File System** [46] (HDFS) is a log dataset collected from a Hadoop-based map-reduce cloud environment using benchmark queries. Each log message contains a timestamp, a log level, and the log content associated with one or multiple block IDs. This dataset contains 11,175,629 log messages. The logs are partitioned into sequences according to the block ID. Each log sequence associated with its unique block ID is labeled normal or abnormal by the Hadoop experts.

• **ThunderBird** [32] is a log dataset collected from a Thunderbird supercomputer system at Sandia National Labs (SBNL) in Albuquerque. Logs are generated on local nodes and collected by a log server. Each log message is an event with an alert label, a timestamp, the source node ID, and the log content. The raw dataset contains 211,212,192 logs. Considering its large size and the training efficiency, we use subset sampling following existing work [51], with a subset of the first 2M log messages, 110,232 of which are alerts.

• **Spirit** [32] is a log dataset collected by Sandia National Labs (SBNL), from a Spirit supercomputer system, sharing the same collection process and log structure with ThunderBird. The raw dataset contains 272,298,969 logs. Similar to ThunderBird, we do subset sampling, with a subset of the first 20M log messages, 9,717,716 of which are alerts.

Data Pre-processing. As described in Section 2.1, we first parse the raw log messages by Drain[14] into structured data with log keys. With structured logs, we employ different partitioning schemes.

Dataset	# of normal	# of abnormal	Anomaly ratio (r^*)	Vocab size
BGL	29,898	2,870	8.7%	1,000
HDFS	575,061	16,838	2.8%	48
ThunderBird	99,026	481	0.48%	638
Spirit	1,181,610	9,291	0.78%	1,075

Table 3. Dataset statistics.

Following existing works [11, 42, 51], the BGL dataset is directly sliced into sequences by sliding windows, while the HDFS dataset is partitioned into sequences by block ID as the labels indicate. For ThunderBird and Spirit datasets, since the logs from different nodes are interleaved with each other, we first partition them by the node ID, then slice the logs of each node into sequences by sliding windows. As in Definition 4, we follow the standard way used by existing works [8, 11, 42] to label abnormal sequences. That is, a log sequence is labeled abnormal if it contains at least one abnormal log key. As the HDFS dataset provides explicit sequence anomaly labels, we use those directly. Table 3 shows the sequence statistics of the datasets after log pre-processing. We do the train-test split with a ratio of 4:6 for the smaller BGL dataset and a ratio of 1:9 for the three larger datasets for training efficiency. All training sets contain anomalies.

8.2 Comparative Methods

We compare Pluto to three groups of comparative methods.

Pattern Mining and Similarity Comparison Methods. This group of methods utilizes pattern mining, similarity comparison, and machine learning techniques to detect anomalies. We compare to the unsupervised methods **PCA** [45], **IsolationForest** [25], **LogCluster** [24], and one-class classifier **OCSVM** [36].

Deep-learning-based Methods. We also compare to deep-learning-based log anomaly detection methods, including **Deeplog** [8], **OC4-Seq** [42], and **LogBert** [11]. With a clean training set without anomalies, these methods aim to learn the pattern of the normal log sequences to detect anomalies during the inference phase. Specifically, Deeplog uses an LSTM architecture to learn the sequential dependencies, by predicting the next log key for the given sequence. OC4Seq leverages an RNN model to map the sequence and its sub-sequences into different latent spaces to detect both global and local level anomalies. Based on the transformer architecture, Logbert applies the masked language model (MLM) task to learn the normal sequence representation, as well as a Volume of Hypersphere Minimization (VHM) task to minimize the hyperspace of the normal sequences in the embedding space.

Sample Selection Methods. Given our proposed solution falls into the category of sample selection, we also compare to sample selection methods widely utilized in learning with noisy labels, including **ITLM** [37], **Co-teaching** [12], and **FINE** [18]. Specifically, Co-teaching maintains two models that select their own small-loss samples and feed each other. ITLM alternates between selecting small-loss samples and re-training the model so that both the model and the selected subset become cleaner. Instead of using the *Small-loss trick*, FINE uses representation as the criteria and recursively selects samples aligning most with the first eigenvector of the datagram matrix. All sample selection methods are deployed with **LogBert**-, which is LogBert without the VHM task as the base log anomaly detection model.

Oracle Method With Clean Training Data. To better interpret the performance degradation caused by polluted training and the effectiveness of sample selection methods, we create clean training sets by removing all anomalies for the four datasets. We train a detector on this clean data using LogBert-, henceforth referred to as **Oracle**, to serve as upper bound in performance.

Implementation Details of Sample Selection Methods. Due to the limited space, we briefly describe the implementation details of the core sample selection methods here, providing full implementation details in the Supplement[3]. The base detector LogBert- for all sample selection methods is built with four layers of Transformer encoders of four attention heads and a hidden size of 256. The learning rate is set to 1e-3 for BGL, ThunderBird, and Spirit datasets and 1e-4 for the HDFS dataset. For methods using loss, like Co-teaching, ITLM, and Pluto, we normalize the sequence loss by the sequence length. For representation-based methods FINE and Pluto, we use the Transformer encoders to generate the sequence embeddings. Then, we apply the most widely used clustering algorithm (K-means) for grouping the data due to its simplicity, lack of tuning requirements, and efficiency for large datasets. We set the cluster number m to 20 for all four datasets and then apply FINE and Pluto to select samples in each cluster. Similar to existing selection methods, such as Co-teaching [12], and ITLM [37], we consider the anomaly (noise) ratio r of Pluto as a hyperparameter. We first set r to the actual anomaly ratio r^* of the respective data sets in Sec. 9.1; later, we evaluate Pluto's sensitivity to r in Sec. 9.5.

Evaluation Metrics. We evaluate the anomaly detection performance with the classification metrics Precision (P), Recall (R), and F-1 score, as well as the threshold-insensitive ranking metric Auc score (Auc).

9 Experimental Results

We first evaluate the overall performance of all approaches (Sec. 9.1), and then display their t-SNE visualization for better result interpretation (Sec. 9.2). We compare selection quality of sample selection approaches in Section 9.3. For our method Pluto, we conduct an ablation study in Section 9.4, analyze its hyperparameter sensitivity in Section 9.5, and evaluate its efficiency in Section 9.7.

9.1 Overall Performance

Table 4 shows the anomaly detection results of all approaches on the four datasets.

As expected, the Oracle outperforms all methods due to its clean training data advantage. Compared to Oracle, its polluted training counterpart, LogBert-, experiences substantial performance degradation on all four datasets, demonstrating the severe negative impact of anomalies in the training set. As for the methods on polluted training data, except for PCA on the HDFS dataset, their overall performance is poor. By comparison, Pluto constantly outperforms other methods in F-1, and achieves the best Auc score on BGL, ThunderBird, and Spirit datasets. Besides Pluto, the second winners in F-1 on the four datasets differ, namely, Deeplog on BGL, PCA on HDFS, OC4Seq on ThunderBird, and OCSVM on Spirit. For four datasets, Pluto outperforms the runner-up by 0.75% (HDFS) to 31.36% (Sprit) absolute increase in F-1.

By comparing Pluto to its base anomaly detection model LogBert-, the sample selection of Pluto brings a significant absolute F-1 increase of 31.39% (BGL) to 69.31% (ThunderBird). Compared to other sample selection methods, Pluto achieves an absolute F-1 increase of 17.45% (BGL) to 68.86% (ThunderBird). While the overall performance of Pluto is good, Pluto does not always achieve the highest precision. This is because the first priority of Pluto is to remove as many as possible anomalies from the training set during the selection to achieve a higher recall after training. Such a strategy may cause some normal samples also to be removed from the training set. This lack of certain normal training samples will lead to a higher false positive rate after training. However, since Pluto brings significant gain in recall in most cases, this trade-off is justified by the overall performance gain.

[3]https://github.com/LeiMa0324/Pluto-SIGMOD25

Table 4. Anomaly detection performance of all baselines. Only baselines with polluted training are considered for metric bolding.

Dataset	BGL				HDFS				ThunderBird				Spirit			
Metric	P (%)	R (%)	F-1 (%)	Auc	P (%)	R (%)	F-1 (%)	Auc	P (%)	R (%)	F-1 (%)	Auc	P (%)	R (%)	F-1 (%)	Auc
Oracle	87.51	96.28	91.69	0.9845	87.28	86.71	86.99	0.9708	82.52	99.65	90.28	0.9983	87.51	96.28	91.69	0.9845
PCA	17.90	11.26	13.82	0.5722	**92.36**	70.31	79.84	**0.9595**	27.12	4.31	7.43	0.8729	8.64	57.28	15.01	0.9398
IsolationForest	64.99	21.76	32.61	0.8125	44.58	68.62	54.04	0.9341	50.96	6.53	11.57	0.9155	12.44	30.37	17.65	0.9518
LogCluster	25.93	0.81	1.58	0.5736	96.72	0.62	1.23	0.4104	20.84	0.11	0.22	0.4734	7.75	0.18	0.36	0.7895
OCSVM	28.42	40.28	33.33	0.7956	8.25	44.30	13.91	0.7908	34.57	15.75	21.64	0.7926	11.46	**85.02**	20.20	0.8498
OC4Seq	29.14	61.52	39.55	0.7705	92.19	33.44	49.08	0.7158	**94.15**	52.85	67.7	0.8795	21.14	11.18	14.63	0.7809
DeepLog	68.63	38.48	49.31	0.8122	57.35	4.13	7.71	0.7029	31.77	4.38	7.71	0.7811	12.75	4.00	6.09	0.9005
LogBert	73.33	19.15	30.37	0.6861	51.17	16.37	24.8	0.8425	4.62	1.04	1.70	0.9561	8.41	4.16	5.57	0.8995
LogBert-	**73.95**	19.44	30.79	0.6813	49.16	15.4	23.45	0.8045	4.76	1.04	1.71	0.9585	8.25	4.09	5.47	0.8939
FINE	44.15	45.33	44.73	0.7674	60.03	35.59	44.69	0.7896	1.20	1.38	1.28	0.8578	4.97	12.03	7.03	0.8853
ITLM	66.93	24.55	35.92	0.6919	64.73	25.21	36.29	0.8100	5.0	1.38	2.16	0.9523	7.80	4.40	5.63	0.8931
Co-teaching	61.03	29.72	39.97	0.7048	45.66	46.05	45.85	0.8981	0.44	1.04	0.62	0.9055	6.12	12.45	8.21	0.8996
Pluto	55.76	**70.28**	**62.18**	**0.8468**	80.85	**80.34**	**80.59**	0.9496	55.17	**99.65**	**71.02**	**0.9977**	**37.59**	81.56	**51.46**	**0.9623**

To fairly compare without the impact of different anomaly thresholds, we vary the top t of log key candidates for all methods using log key prediction, including Deeplog, the Logbert variants, and the sample selection methods using LogBert-, including FINE, ITLM, Co-teaching and Pluto. Figure 8 presents the F-1 and Auc scores of the above approaches under different t. Pluto significantly outperforms the baselines in F-1 under different top t, except in ThunderBird dataset with $t \leq 10$. Similarly, Pluto also constantly outperforms the baselines in Auc score under different t. We also make an interesting observation that, with t increasing, the performance of most baselines either stays low or decreases. However, the performance of Pluto stays relatively stable for BGL, HDFS, and ThunderBird datasets. This indicates a stronger anomaly detection ability, even under a loose anomaly criterion.

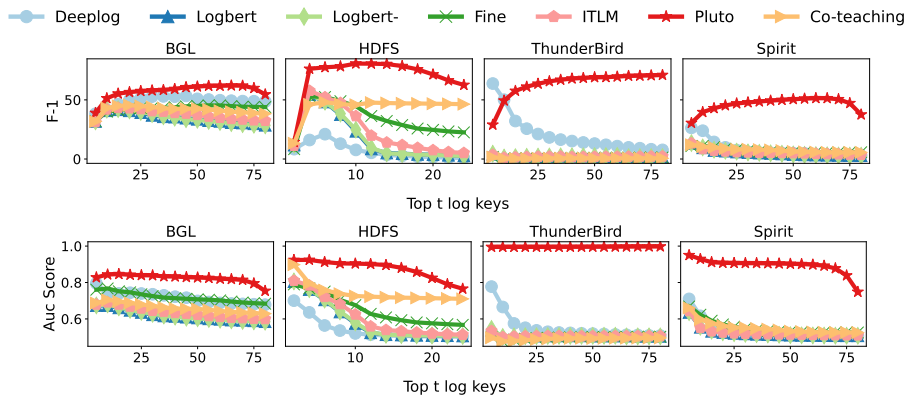


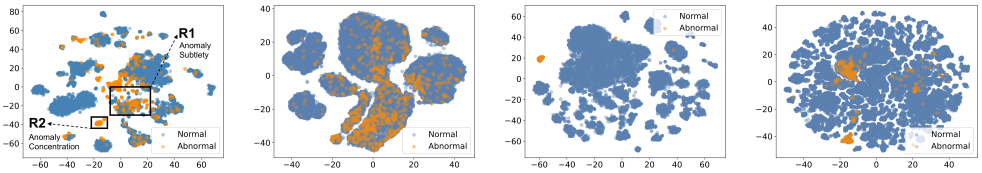
Fig. 8. Performance vs. Top t log key candidates.

We observe a common trend that most approaches perform better on the BGL and HDFS datasets than the ThunderBird and Spirit datasets. We believe such performance differences are caused by the different pollution distributions of the four datasets, which will be discussed with our t-SNE visualization later in Section 9.2.

9.2 Dataset Visualization

Figures 9(a) to 9(d) show the t-SNE visualization of the sequence embedding of the four polluted training sets (before selection). It reveals anomaly-related observations consistent with the motivation example in Figure 1. When normal patterns are diverse in the dataset, the embedding spaces are subject to imbalanced pollution with varying degrees. While BGL and HDFS datasets have more

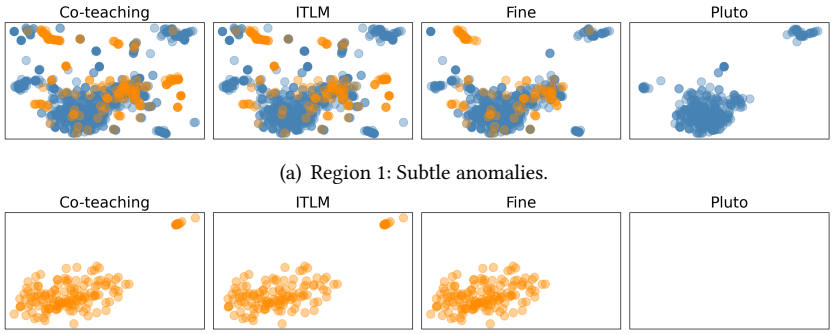
scattered pollution, the pollution in ThunderBird and Spirit datasets is extremely concentrated — indicating strong abnormal patterns. Compared to random pollution, anomaly detection models are more vulnerable to these collective and concentrated anomalies in the training set. This explains the universally poor performance of the baselines on ThunderBird and Spirit datasets.



(a) BGL dataset (b) HDFS dataset (c) ThunderBird dataset (d) Spirit dataset

Fig. 9. t-SNE visualization of sequence embeddings of the four training sets. a) R1: region of anomaly subtlety (Embedding overlapping between normal samples and anomalies). R2: region of anomaly concentration (Clustered anomalies).

Selection with Subtle or Concentrated Anomalies. We evaluate how the selection methods Co-teaching, ITLM, FINE, and Pluto perform with data sets containing anomaly subtlety and anomaly concentration (Sec. 1). In Figure 9(a) of the BGL dataset, we highlight two regions, R1 and R2, with subtle and concentrated anomalies, respectively. Figure 10 shows the selection results of the four selection methods in the two regions. While all baseline selection methods fail to clean either R1 or R2 effectively, Pluto successfully achieves clean selection in both regions due to its high pollution identification (Sec. 5), spectrum-purifying selection strategy (Sec. 6) and refinement (Sec. 7), which we further analyze in the next subsection (Sec.9.3).



(a) Region 1: Subtle anomalies.

(b) Region 2: Concentrated anomalies.

Fig. 10. Selection results with anomaly subtlety and anomaly concentration.

9.3 Subset Selection Quality

Selection Quality. We examine the effectiveness of selection by the sample selection methods FINE, ITLM, Co-teaching, and Pluto for all four datasets. FINE recursively selects samples from its last selected subset, while Co-teaching selects small-loss samples in every batch. ITLM and Pluto alternate between selecting and training the model on the last selected subset for certain epochs. For a fair comparison, we choose the first subset selected by FINE and the last subsets selected by Co-teaching, ITLM, and Pluto.

Figure 11 shows the number of normal and abnormal sequences in the original polluted training data and the subsets selected by all sample selection methods. Pluto achieves the best selection quality overall, with a similar number of normal sequences and the fewest anomalies selected for all datasets. Through sample selection, keeping 77.7% (BGL) to 96.6% (ThunberBird) of the normal

sequences, Pluto effectively removes 90.3% (BGL) to 100.0% (ThunderBird, HDFS) of the anomalies from the original dataset. This explains its significant performance gain over other methods in Table 4. Such good selection quality comes from the three stages of the Pluto pipeline: high pollution identification (Sec.5), spectrum-purifying selection (Sec. 6) and iterative refinement (Sec. 7). We will examine them in a stage-by-stage manner later in this subsection.

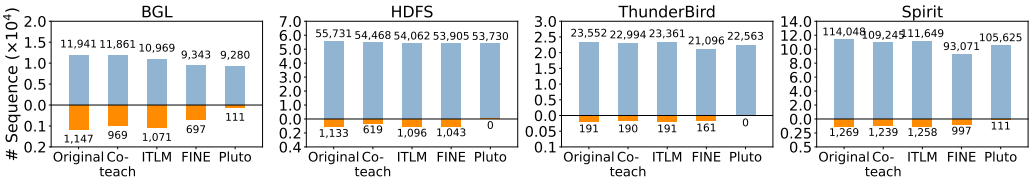


Fig. 11. Selection quality of different sample selection methods. Blue bars: # of normal samples. Orange bars: # of abnormal samples.

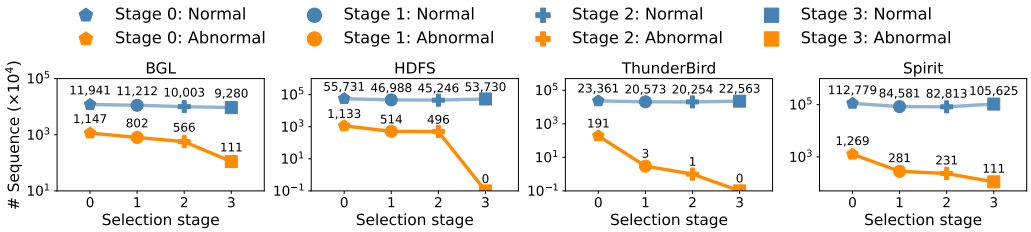


Fig. 12. Effectiveness of Pluto selection stages. Stage 0: Original dataset. Stage 1: After discarding high pollution. Stage 2: After selection. Stage 3: After refinement.

FINE is generally the worst method, selecting (roughly) the most anomalies. Unlike Pluto that distinguishes the pollution level for each cluster, FINE selects samples close to the first singular vector in all clusters. This universe selection strategy cannot reduce the anomaly ratio in highly polluted clusters. It compensates for this by employing a recursive selection strategy so that the next subset selected from the current one is cleaner. By leveraging the memorization effect [2, 50], the model using FINE is robust to scattered anomalies in the early stage. However, its compensation barely works on ThunderBird and Spirit datasets containing concentrated anomalies that can quickly cause anomaly overfitting.

In contrast to FINE, Co-teaching has a better selection quality but does not achieve a good performance. This may be because Co-teaching selects and trains at the same time. With the concentrated anomalies in the highly polluted regions, the model quickly overfits to anomalies, compromising the effectiveness of the *small-loss trick*.

Effectiveness of Pluto Selection Stages. We focus on the contribution of each of the stages in the Pluto pipeline in progressively improving the selected subset. Figure 12 shows the number of normal and abnormal sequences of four sets throughout the Pluto selection process, *i.e.*, the original dataset, intermediate subset after discarding the highly polluted clusters (Sec. 5), intermediate subset after selection (Sec. 6), and final subset after refinement (Sec. 7), respectively. Following these selection stages, we observe a stable trend in the number of normal sequences and a constant decrease in the number of abnormal ones. Specifically, the comparison between stages 0 and 1 shows that simply discarding the highly polluted clusters removes 30.1% (BGL) to 98.4% (ThunderBird) of anomalies, validating the utility of the pollution quantification theory (Sec. 4), and the effectiveness of the high pollution identification (Sec. 5). The selection strategy (Sec. 6) in Stage 2 helps remove more anomalies from slightly polluted clusters. Lastly, the subset refinement (Sec. 7) in Stage 3

contributes another significant anomaly reduction of 51.9% (Spirit) to 100.0% (HDFS, ThunderBird) compared to stage 2. This anomaly reduction and the resulting gain in the final performance can be worth the additional computational costs of conducting these iterative refinement steps. By combining the above components, Pluto progressively provides a superior sample selection performance.

9.4 Ablation Study

In this section, we examine the relative contributions of high pollution identification and subset refinement to the final anomaly detection performance. The high pollution identification contributes to discarding highly polluted clusters, while the subset refinement further cleans the subset iteratively. We design two Pluto variants to evaluate the contribution of each component: Pluto^{-H} corresponds to Pluto without the identification of the highly polluted clusters (Sec. 5), which applies the selection strategy for all clusters. Pluto^{-R} applies a one-time Pluto base selection without the subset refinement (Sec. 7). Table 5 shows the anomaly detection results of all three Pluto variants, including Pluto itself. Pluto achieves the best Auc score on all datasets and the best F-1 on BGL and HDFS datasets. By comparing Pluto^{-H} and Pluto^{-R} with Pluto, we observe that the components contribute differently for different datasets. For BGL, while both the high pollution identification and subset refinement contribute to the final Pluto performance, the former brings a larger F-1 improvement of 21.91% compared to 5.66% brought by the latter. However, for the HDFS dataset, most of the performance improvement comes from subset refinement, contributing an F-1 gain of 36.17%. Meanwhile, for ThunderBird and Spirit datasets, due to their extremely concentrated anomalies as shown in Figure 9, the high pollution identification contributes the most towards the final performance. Although we show that refinement indeed further reduces the number of anomalies for all datasets in Figure 12, bringing an absolute increase of 0.34% (ThunderBird) to 2.24% (Spirit) in Recall, the subset refinement also slightly removes some normal sequences, causing an absolute decrease of 1.54% (Spirit) to 1.89% (ThunderBird) in Precision. We consider this acceptable in anomaly detection scenarios especially where recall is more essential.

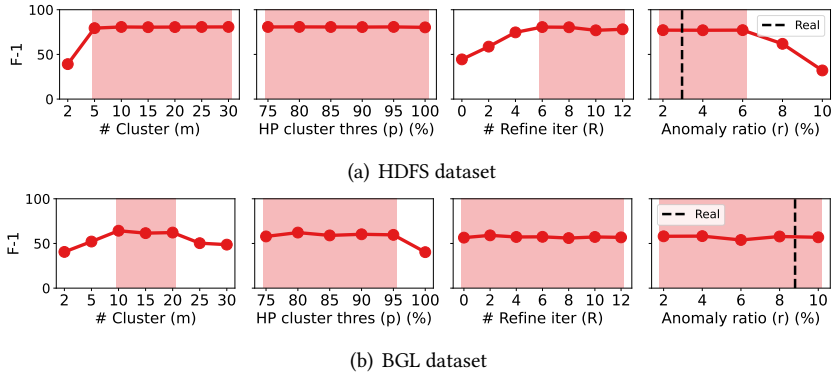
Table 5. Ablation study of Pluto.

Dataset	BGL				HDFS				ThunderBird				Spirit			
Metric	P (%)	R (%)	F-1 (%)	Auc	P (%)	R (%)	F-1 (%)	Auc	P (%)	R (%)	F-1 (%)	Auc	P (%)	R (%)	F-1 (%)	Auc
Pluto^{-H}	41.11	39.47	40.27	0.6655	78.67	81.92	80.26	0.9068	3.23	1.38	1.93	0.5052	5.33	9.74	6.89	0.5387
Pluto^{-R}	70.35	47.24	56.52	0.7271	58.54	35.79	44.42	0.6767	57.06	99.31	**72.48**	0.9913	39.13	79.32	**52.41**	0.9428
Pluto	55.76	70.28	**62.18**	**0.8468**	80.85	80.34	**80.59**	**0.9496**	55.17	99.65	71.02	**0.9997**	37.59	81.56	51.46	**0.9623**

9.5 Hyperparameter Sensitivity Study

Now we evaluate Pluto's sensitivity to hyperparameters on the more popular datasets used in the recent works, namely BGL and HDFS [8, 16, 42, 51]. The hyperparameters include (1) cluster number m ranging from 2 to 30, (2) percent threshold of the highly polluted clusters p from 75% to 100%, (3) number of refinement iterations R ranging from 0 to 12, and (4) given anomaly ratio r ranging from 2% to 10%. Figure 13 shows the F-1 results of Pluto with different hyperparameter values. We observe that Pluto can be sensitive to extreme hyperparameter values for cluster number, refinement iterations, or anomaly ratios too far away from the actual anomaly ratio. In conjunction with the ablation study, we see that Pluto is more sensitive to the hyperparameters associated with the more influential components. For example, Pluto is more sensitive to the refinement-related hyperparameter R on HDFS since, for this dataset, the subset refinement contributes most to the performance gain (per results of the ablation study in Table 5). Similarly, Pluto is more sensitive

to high-pollution-identification-related hyperparameters m and p on the BGL dataset as, in this case, high pollution identification contributes more to the performance gain.



(a) HDFS dataset

(b) BGL dataset

Fig. 13. Hyperparameter sensitivity of Pluto. Shadow: performance stable area. Dashed line: real anomaly ratio of the dataset.

9.6 Low Anomaly Ratio Data Sets

We now test how Pluto performs on datasets with very low anomaly ratios. Given ThunderBird, the dataset with the lowest anomaly ratio (0.48%) among our four datasets, we create new datasets with smaller anomaly ratios of 0.05% and 0.10% by removing a portion of the original anomalies while keeping all its normal sequences. Figure 14 shows the performance of four sample selection methods, together with the basic detector LogBert- on these altered datasets. We observe that Pluto still results in a significant performance gain compared to the baselines for these lower anomaly ratios. Since Pluto targets the characteristics of anomaly concentration and anomaly subtlety, this indicates that Pluto is able to identify the anomalies in the training data as long as they show these characteristics, even under an extremely low anomaly ratio.

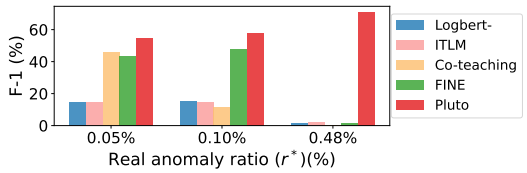
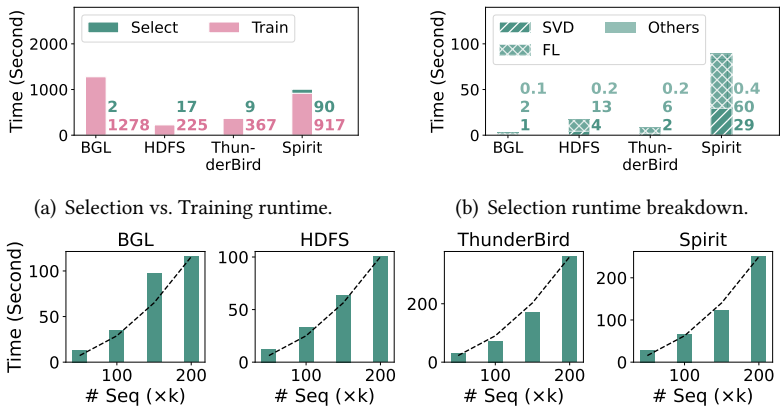


Fig. 14. Anomaly detection performance of baselines on ThunderBird dataset with lower (real) anomaly ratios.

9.7 Runtime Efficiency

We assess the selection efficiency of Pluto in Figure 15. During each selection iteration, Pluto performs a selection after G epochs of training. Thus, in Figure 15(a), we report the average runtime of one Pluto subset selection versus the average training time of G epochs. The Pluto selection runtime is relatively small, namely, 0.02% (BGL) to 9.7% (Spirit) of the training time. We breakdown the selection runtime for different components, including the runtime of (1) SVD, (2) GreedyFacilityLocation and (3) others as in Algo. 1 (Sec. 6). Consistent with our complexity analysis in Algo. 1, Figure 15(b) shows that the selection runtime is dominated by SVD and GreedyFacilityLocation algorithms. We evaluate the selection runtime versus the number of



(a) Selection vs. Training runtime.

(b) Selection runtime breakdown.

(c) Selection runtime vs. sequence number.

Fig. 15. Runtime Efficiency of Pluto. a) Selection runtime is small compared to training runtime. b) Selection runtime is dominated by the cost of SVD and FL algorithms. c) The selection runtime approximately matches a quadratic function (dashed line) in the number of sequences.

sequences ranging from 50K to 200K. Again consistent with our complexity analysis in Algo. 1, Figure 15(c) confirm that the selection runtime is approximately quadratic in the number of sequences.

10 Related Work

Log Anomaly Detection Strategies. The research on log anomaly detection can be categorized into three groups [19, 20]: unsupervised, supervised, and semi-supervised, according to the availability of the anomaly labels. By leveraging traditional models like Support Vector Machine, Linear Regression, or Decision Tree, supervised methods [1, 15, 23] utilize the anomaly labels to learn a binary classifier to detect anomalies. Without any labels, unsupervised methods like LOF [3], IsolationForest [25], and PCA [45] separate the sequences based on their distance, under the assumption that abnormal sequences are far away from the normal sequences or have a smaller density [4].

Due to the imbalance class distribution and the lack of anomaly labels, most recent works [8, 11, 16, 22, 42, 51] adopt a semi-supervised approach using deep learning, aiming to train a one-class classifier of what is considered normal data using a training set composed of only normal sequences. Based on LSTM architecture, DeepLog [8] learns the normal sequence pattern by predicting the next log key of a given subsequence. Methods like LogBert [11], CAT [51], and UCAD [22], on the other hand, learn the normal representation by the Transformer architecture or attention mechanism, leveraging a one-class objective or log semantics. Although the one-class classifiers achieve good performance on anomaly detection, a recent survey [20] points out that even slight pollution in the training set can cause severe performance degradation for the semi-supervised methods. This now is our focus.

Sample Selection in Learning with Noisy Data. Sample selection in learning with data with noisy labels (i.e., with corrupted labels) is a research domain closely related to our problem [18, 40, 43]. Given a training set with noisy labels, this line of research focuses on selecting clean samples with the correct labels typically *for classification.* Under the assumption of balanced classes and small random pollution in each class, the clean samples with correct labels are considered inliers, and noisy samples with wrong labels are considered outliers. Empirical studies [2, 39] observe

the *Memorization Effect* that the model learns clean patterns first and then later overfits to noisy patterns. Based on such observations, the *Small-loss trick* is commonly employed by recent works [6, 12, 17, 37, 49] to select clean samples. However, the anomaly concentration in the unlabeled log data violates the random pollution assumption, where similar anomalies can be easily learned by the model with a small loss. Thus, the effectiveness of these methods is limited.

Instead of relying on loss, another group of methods [18, 43, 44] leverages the data representation to design the selection criteria. For this, they assume certain topological structures of the clean samples in the high-order latent space. However, similar to the *Small-loss trick* methods, the concentrated anomalies in the unlabeled log data can form a strong local topological structure, which will still compromises the effectiveness of the representation-based methods.

11 Conclusion

Given a log sequence dataset polluted with anomalies, Pluto selects a clean representative subset as the new training set to train the anomaly detection model. Given the sequence embeddings of the polluted dataset, Pluto first partitions the embedding space into regions and accurately identifies and discards the highly polluted regions using our pollution level estimator. Then, Pluto applies the *Spectrum-purifying* selection strategy on the remaining slightly polluted regions, selecting samples that best purify the eigenvector spectrum. Pluto further filters and refines the above selected *base subset* by iteratively alternating between selection, subset filtering, model training on the selected subset, and source data filtering, leveraging dynamic data artifacts generated by the model. At last, Pluto retrains the anomaly detection model on the final *refined subset*. Our experiments demonstrate that Pluto significantly removes 90.3% to 100.0% of the anomalies, bringing an absolute F-1 improvement of up to 68.86% compared to state-of-the-art sample selection on real-world log data sets.

While we explore settings without the availability of any labels, future work could explore how best to leverage either a limited small number of ground-truth labels or many noisy labels by boosting, selecting clean or refurbishing the noisy labels.

Acknowledgments

This work is supported in part by NSF grants IIS-1815866, IIS-1910880, CSSI-2103832, CNS-1852498, NRT-HDR-2021871, and U.S. Dept. of Edu. P200A180088. Lei Cao is supported by the NSF (DBI-2327954) and Amazon Research Award. Thanks also to the members of Daisy research group who have given valuable feedback on this project.

References

[1] Shivam Agarwal. 2013. Data mining: Data mining concepts and techniques. In *2013 International Conference on Machine Intelligence and Research Advancement*. IEEE, 203–207.

[2] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. 2017. A closer look at memorization in deep networks. In *International Conference on Machine Learning*. PMLR, 233–242.

[3] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. LOF: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*. 93–104.

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

[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2010. Anomaly detection for discrete sequences: A survey. *IEEE transactions on knowledge and data engineering* 24, 5 (2010), 823–839.

[6] Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang. 2019. Understanding and utilizing deep neural networks trained with noisy labels. In *International Conference on Machine Learning*. PMLR, 1062–1070.

[7] Min Du and Feifei Li. 2016. Spell: Streaming parsing of system event logs. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 859–864.

[8] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. 2017. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*. 1285–1298.

[9] Ronald A Fisher. 1936. The use of multiple measurements in taxonomic problems. *Annals of eugenics* 7, 2 (1936), 179–188.

[10] Gene H Golub and Charles F Van Loan. 2013. *Matrix computations*. JHU press.

[11] Haixuan Guo, Shuhan Yuan, and Xintao Wu. 2021. Logbert: Log anomaly detection via bert. In *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[12] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems* 31 (2018).

[13] Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. 2022. Adbench: Anomaly detection benchmark. *Advances in Neural Information Processing Systems* 35 (2022), 32142–32159.

[14] Pinjia He, Jieming Zhu, Zibin Zheng, and Michael R Lyu. 2017. Drain: An online log parsing approach with fixed depth tree. In *2017 IEEE international conference on web services (ICWS)*. IEEE, 33–40.

[15] Shilin He, Jieming Zhu, Pinjia He, and Michael R Lyu. 2016. Experience report: System log analysis for anomaly detection. In *2016 IEEE 27th international symposium on software reliability engineering (ISSRE)*. IEEE, 207–218.

[16] Peng Jia, Shaofeng Cai, Beng Chin Ooi, Pinghui Wang, and Yiyuan Xiong. 2023. Robust and Transferable Log-based Anomaly Detection. *Proceedings of the ACM on Management of Data* 1, 1 (2023), 1–26.

[17] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. MentorNet: Learning Data-Driven Curriculum for Very Deep Neural Networks on Corrupted Labels. arXiv:1712.05055 [cs.CV]

[18] Taehyeon Kim, Jongwoo Ko, JinHwan Choi, Se-Young Yun, et al. 2021. Fine samples for learning with noisy labels. *Advances in Neural Information Processing Systems* 34 (2021), 24137–24149.

[19] Max Landauer, Sebastian Onder, Florian Skopik, and Markus Wurzenberger. 2023. Deep learning for anomaly detection in log data: A survey. *Machine Learning with Applications* 12 (2023), 100470.

[20] Van-Hoang Le and Hongyu Zhang. 2022. Log-based anomaly detection with deep learning: How far are we?. In *Proceedings of the 44th International Conference on Software Engineering*. 1356–1367.

[21] Kimin Lee, Sukmin Yun, Kibok Lee, Honglak Lee, Bo Li, and Jinwoo Shin. 2019. Robust inference via generative classifiers for handling noisy labels. In *International conference on machine learning*. PMLR, 3763–3772.

[22] Sainan Li, Qilei Yin, Guoliang Li, Qi Li, Zhuotao Liu, and Jinwei Zhu. 2022. Unsupervised Contextual Anomaly Detection for Database Systems. In *Proceedings of the 2022 International Conference on Management of Data* (Philadelphia, PA, USA) *(SIGMOD '22)*. Association for Computing Machinery, New York, NY, USA, 788–802. https://doi.org/10.1145/3514221.3517861

[23] Yinglung Liang, Yanyong Zhang, Hui Xiong, and Ramendra Sahoo. 2007. Failure prediction in ibm bluegene/l event logs. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*. IEEE, 583–588.

[24] Qingwei Lin, Hongyu Zhang, Jian-Guang Lou, Yu Zhang, and Xuewei Chen. 2016. Log clustering based problem identification for online service systems. In *Proceedings of the 38th International Conference on Software Engineering Companion*. 102–111.

[25] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *2008 eighth ieee international conference on data mining*. IEEE, 413–422.

[26] Matthias Löffler, Anderson Y Zhang, and Harrison H Zhou. 2021. Optimality of spectral clustering in the Gaussian mixture model. *The Annals of Statistics* 49, 5 (2021), 2506–2530.

[27] Carl D Meyer and Gilbert W Stewart. 1988. Derivatives and perturbations of eigenvectors. *SIAM J. Numer. Anal.* 25, 3 (1988), 679–691.

[28] Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, J. Stoer (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 234–243.

[29] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. 2015. Lazier than Lazy Greedy. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (Austin, Texas) *(AAAI'15)*. AAAI Press, 1812–1818.

[30] Baharan Mirzasoleiman, Kaidi Cao, and Jure Leskovec. 2020. Coresets for robust training of deep neural networks against noisy labels. *Advances in Neural Information Processing Systems* 33 (2020), 11465–11477.

[31] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functions—I. *Mathematical programming* 14 (1978), 265–294.

[32] Adam Oliner and Jon Stearley. 2007. What supercomputers say: A study of five system logs. In *37th annual IEEE/IFIP international conference on dependable systems and networks (DSN'07)*. IEEE, 575–584.

[33] Samet Oymak, Zalan Fabian, Mingchen Li, and Mahdi Soltanolkotabi. 2019. Generalization guarantees for neural networks via harnessing the low-rank structure of the jacobian. *arXiv preprint arXiv:1906.05392* (2019).

[34] 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* 109, 5 (2021), 756–795.

[35] Ville Satopaa, Jeannie Albrecht, David Irwin, and Barath Raghavan. 2011. Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior. In *2011 31st International Conference on Distributed Computing Systems Workshops*. 166–171. https://doi.org/10.1109/ICDCSW.2011.20

[36] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. 2001. Estimating the support of a high-dimensional distribution. *Neural computation* 13, 7 (2001), 1443–1471.

[37] Yanyao Shen and Sujay Sanghavi. 2019. Learning with bad training data via iterative trimmed loss minimization. In *International Conference on Machine Learning*. PMLR, 5739–5748.

[38] Tao Shi, Mikhail Belkin, and Bin Yu. 2008. Data spectroscopy: Learning mixture models using eigenspaces of convolution operators. In *Proceedings of the 25th international conference on Machine learning*. 936–943.

[39] Hwanjun Song, Minseok Kim, Dongmin Park, and Jae-Gil Lee. 2019. How does early stopping help generalization against label noise? *arXiv preprint arXiv:1911.08059* (2019).

[40] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. 2022. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[41] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605. http://jmlr.org/papers/v9/vandermaaten08a.html

[42] Zhiwei Wang, Zhengzhang Chen, Jingchao Ni, Hui Liu, Haifeng Chen, and Jiliang Tang. 2021. Multi-scale one-class recurrent neural networks for discrete event sequence anomaly detection. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*. 3726–3734.

[43] Pengxiang Wu, Songzhu Zheng, Mayank Goswami, Dimitris Metaxas, and Chao Chen. 2020. A topological filter for learning with label noise. *Advances in neural information processing systems* 33 (2020), 21382–21393.

[44] Zhi-Fan Wu, Tong Wei, Jianwen Jiang, Chaojie Mao, Mingqian Tang, and Yu-Feng Li. 2021. NGC: A Unified Framework for Learning With Open-World Noisy Data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 62–71.

[45] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael Jordan. 2009. Largescale system problem detection by mining console logs. *Proceedings of SOSP'09* (2009).

[46] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I Jordan. 2009. Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*. 117–132.

[47] Quanming Yao, Hansi Yang, Bo Han, Gang Niu, and James Tin-Yau Kwok. 2020. Searching to exploit memorization effect in learning with noisy labels. In *International Conference on Machine Learning*. PMLR, 10789–10798.

[48] Xuanyu Yi, Kaihua Tang, Xiansheng Hua, Joo Hwee Lim, and Hanwang Zhang. 2022. Identifying Hard Noise in Long-Tailed Sample Distribution. In *European Conference on Computer Vision*.

[49] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. 2019. How does disagreement help generalization against label corruption?. In *International Conference on Machine Learning*. PMLR, 7164–7173.

[50] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Sy8gdB9xx

[51] Shengming Zhang, Yanchi Liu, Xuchao Zhang, Wei Cheng, Haifeng Chen, and Hui Xiong. 2022. CAT: Beyond Efficient Transformer for Content-Aware Anomaly Detection in Event Sequences. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Washington DC, USA) *(KDD '22)*. Association for Computing Machinery, New York, NY, USA, 4541–4550. https://doi.org/10.1145/3534678.3539155

[52] Tianyi Zhou, Shengjie Wang, and Jeff Bilmes. 2021. Robust curriculum learning: from clean label detection to noisy label self-correction. In *International Conference on Learning Representations*.

Received January 2024; revised April 2024; accepted May 2024