AutoKAD: Empowering KPI Anomaly Detection with Label-Free Deployment

Zhaoyang Yu[1,5], Changhua Pei[2*], Shenglin Zhang[3,6], Xidao Wen[4], Jianhui Li[2], Gaogang Xie[2], Dan Pei[1,5]
[1]Tsinghua University, yu-zy20@mails.tsinghua.edu.cn, peidan@tsinghua.edu.cn
[2]Computer Network Information Center, Chinese Academy of Sciences, {chpei, lijh, xie}@cnic.cn
[3]Nankai University, zhangsl@nankai.edu.cn
[4]BizSeer Technology, China, mwen8103@gmail.com
[5]Beijing National Research Center for Information Science and Technology
[6]Haihe Laboratory of Information Technology Application Innovation

***Abstract*—Monitoring Key Performance Indicators (KPIs) and detecting anomalies in online service systems is critical. However, choosing the right KPI anomaly detection algorithm and appropriate hyperparameters presents a challenge. Conventional Automated Machine Learning (AutoML) struggles to address this because the hold-out dataset lacks labels and its loss doesn't reliably reflect anomaly detection accuracy. To address the above challenges, this paper introduces *AutoKAD*, an AutoML framework designed to solve the combined algorithm selection and hyperparameter optimization problem for unsupervised KPI Anomaly Detection. We propose a label-free universal objective function, inspired by the Local Outlier Factor (LOF), for evaluating AutoML trials. Additionally, we improve the acquisition function and designs a cluster-based warm start strategy to enhance exploration effectiveness and efficiency. The experimental results on three real-world datasets show that our approach outperforms the SOTA model selection algorithm by 11% in F1-score and achieves comparable performance (99%) with theoretically optimal results. We believe that *AutoKAD* can greatly improve the deployment feasibility of existing anomaly detection algorithms in real-world systems. Our code is anonymously released at https://github.com/NetManAIOps/AutoKAD.**

I. Introduction

In today's digital world, online service systems, such as search engines, e-commerce platforms, and social networks, have become an integral part of our daily lives. To ensure seamless service and maintain user satisfaction, IT operations engineers in these companies closely monitor Key Performance Indicators (KPIs) such as response time and success rate, providing a comprehensive overview of the system's performance. KPI anomaly detection (KAD) plays a crucial role in identifying potential issues by detecting anomalies in KPIs, thereby accelerating the process of failure diagnosis and mitigation.

Tremendous efforts have been devoted to KPI anomaly detection in the literature. Existing work includes supervised approaches based on ensemble learning (*e.g.*, Opprentice [1], EGADS [2]), semi-supervised approaches (*e.g.*, ADS [3], ATAD [4]), and unsupervised approaches, which involve traditional statistical methods (*e.g.*, Holt-winters [5], ARIMA [6]) and deep learning-based methods (*e.g.*, Donut [7], LSTM-NDT [8], Buzz [9], OmniAnomaly [10], Interfusion [11], AnomalyTrans [12]). Unfortunately, faced with various algorithms and numerous KPIs in the real world, we encounter a classic challenge for machine learning algorithms, a *Combined Algorithm Selection and Hyperparameter optimization* (CASH) problem [13]. More specifically, the first one is *how to select suitable algorithms*. Although the superiority of each algorithm has been illustrated in its paper, the performance on different KPIs with various patterns (*e.g.*, seasonal, variational, and stationary) are unstable. For example, Donut is designed only for seasonal KPIs and performs poorly on other KPIs. The second one is *hyperparameter tuning*. Deep learning-based algorithms have been widely applied to unsupervised KPI anomaly detection to enhance performance in recent years. However, some hyperparameters in neural networks should be configured in advance. For example, the F1-score of Donut could range from 0.2 to 0.9 even on the same KPI given different settings of the hyperparameters. In summary, algorithm selection and hyperparameter selection could directly influence anomaly detection accuracy.

In practice, selecting suitable algorithms and tuning parameters is difficult for operators for the following reasons. First, most traditional operators are responsible for monitoring rather than being machine learning experts. The selection of algorithms and hyperparameters is a black-box task for them. Second, the number of KPIs of a typical service in the real world can be tremendous, often more than tens of thousands. Manually selecting a satisfying algorithm and hyperparameter for each KPI is time-consuming. Third, the search space of hyperparameters is usually huge. Sometimes it is hard for a human to find the best algorithm and hyperparameter combination. The complexity of algorithm selection and hyperparameter optimization slows down the pace of algorithms' deployment in the production environment. Consequently, designing an automated framework to solve the CASH problem for KAD for operators is in urgent demand.

AutoML (Automated machine learning) has been a popular method to deal with the CASH problem in the field of machine learning [13]–[22]. However, directly applying existing AutoML frameworks faces the following challenges. 1) *Unattainable objective function*. Considering it is difficult to obtain sufficient anomaly labels, it is unfeasible to accurately

*Changhua Pei is the corresponding author.

evaluate the performance of a specific AutoML selection by widely used evaluation metric (*e.g.,* F1-score). In addition, the performance of anomaly detection algorithms cannot be effectively demonstrated using the validation loss on hold-out datasets. This is because that existing popular KPI anomaly detection methods [7]–[11] aim to model the pattern of input KPI based on reconstruction or prediction loss. A smaller loss on the hold-out dataset can not represent a better result for anomaly detection as the input is noisy and may contain anomalies. 2) *High time complexity*. As mentioned earlier, the number of KPIs in practice is huge. The company is unwilling to bear the cost of time and resources if the AutoML model takes too long to search for algorithm and hyperparameter combinations.

To address the above two challenges, in this paper, we propose *AutoKAD*, an AutoML framework designed to solve the CASH problem for unsupervised KPI Anomaly Detection. *AutoKAD* consists of three phases: *cluster-based warm start*, *Bayesian Optimization (BO)-based configuration search*, and *rule-based configuration recommendation* as illustrated in Fig. 1. In *BO-based configuration search* phase, *AutoKAD* proposes a novel label-free objective function called MSE-NF (Mean Squared Error with Normal Factor) to assess the effectiveness of a specific trial given by AutoML. In the MSE part, we measure the similarity between the output KPI and the input KPI. However, since the input KPI is noisy and may contain anomalies, relying completely on the MSE part will lead to an overfitting of abnormalities in the input. Inspired by the **L**ocal **O**utlier **F**actor (LOF) [23], we propose NF to evaluate the normality of reconstructed KPIs.

To tackle the challenge of time efficiency, *AutoKAD* firstly activates the clustering-based warm start mechanism to learn the successful experience of similar KPIs tuned in the past (shown as *cluster-based warm start* in Fig. 1). This phase dramatically increases the time efficiency of *AutoKAD*. Moreover, we optimize the acquisition function of Bayesian Optimization, specifically by improving the original Expected Improvement (EI) acquisition function to derive the Similarity Weighted Expected Improvement (SW-EI), thereby enhancing the effectiveness of configuration search. Finally, our rule-based strategy recommends appropriate configurations returned by *AutoKAD* according to the preferences of operators.

The contributions of this paper are summarized as follows:

- We propose *AutoKAD*, an AutoML framework for unsupervised KPI Anomaly Detection, which assists operators in choosing better algorithms and optimal parameters. The comprehensive evaluation shows that *AutoKAD* achieves an F1-score of 0.80 for each KPI on our datasets, which is up to 40% higher than a widely used Donut model [7] with carefully tuned hyperparameters.
- Through the well-designed evaluation objective function MSE-NF and acquisition function SW-EI, for the first time, we have successfully overcome the challenge of evaluating unsupervised KPI anomaly detection tasks without the assistance of labels.
- A cluster-based warm start strategy and a configuration recommendation strategy are proposed by *AutoKAD* to improve the exploration efficiency and effectiveness. Our cluster-based warm start mechanism reduces the searching time of *AutoKAD* from one hour to 15 minutes. Our strategy of configuration recommendation mechanism can give close-to-optimal configurations, according to operators' preferences.
- We have deployed *AutoKAD* in a real-world online service system in a top commercial bank serving their KAD models. Our code is anonymously released at https://github.com/NetManAIOps/AutoKADq and we believe that *AutoKAD* can greatly advance the deployment of anomaly detection algorithms in practical systems.

II. Background

A. KPI and KPI Anomaly Detection

Key Performance Indicators (KPIs) are time-series data collected from a myriad of sources such as network traces, web access logs, and data centers, among others. As a substantial gauge of service quality, KPI data often provide crucial information. A consistent KPI typically reflects the smooth operation of a service. However, anomalies, such as spikes or dips, within the KPIs may indicate potential service faults. Fig. 2 shows three typical KPIs: seasonal, stable, and unstable, with anomalies marked by red dots. Since most user behaviors are cyclical, the seasonal type of KPI is usually the majority in the real-world environment [7]. Anomalies within KPIs could lead to a significant influence on the revenue of the company. Therefore, it is essential for the company to closely monitor KPIs and promptly detect anomalies in KPIs to reduce unexpected economic losses.

In recent years, dozens of KPI anomaly detection algorithms based on machine learning have been proposed to help the operators identify the anomalies and achieve great success [1], [2], [5]–[9], [12], [24]. Because of the difficulty of obtaining high-quality labels of KPIs, unsupervised KPI anomaly detection algorithms are more popular than supervised algorithms in this field. Most unsupervised KPI anomaly detection algorithms are based on prediction or reconstruction. The algorithms based on prediction or reconstruction take an interval of data points from the KPI as input. The prediction algorithms use the input to predict the next data point. The reconstruction algorithms use the input to reconstruct the whole interval. The unsupervised KPI anomaly detection algorithms are trained with normal KPI data and learn normal patterns. In the testing phase, the algorithms attempt to predict or reconstruct the normal KPI. If the prediction or reconstruction KPI is very different from the original KPI, the algorithm declares that an anomaly happens. Both prediction algorithms and reconstruction algorithms can estimate the normal KPI from the raw KPI data.

B. AutoML

Owing to the remarkable superiority in anomaly detection performance that machine learning-based models offer compared to traditional methods, there has been a growing trend

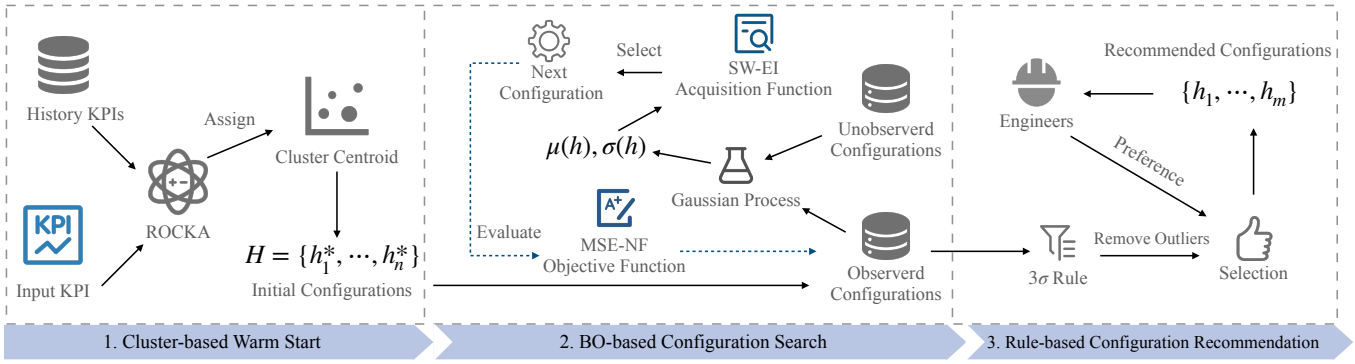
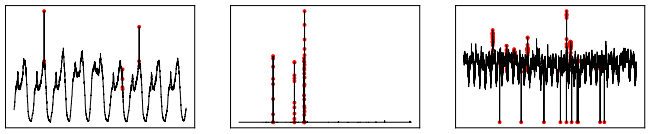


Fig. 1. Overall architecture of *AutoKAD*



(a) seasonal (b) stable (c) unstable

Fig. 2. Examples of three typical KPIs (seasonal, stable, unstable) from global Internet companies. The red dots mark the obvious anomalies.

of deploying machine learning-based models in industrial environments. However, it's noteworthy that the majority of operators in the industry, tasked with anomaly detection, are non-experts. They often resort to utilizing these machine learning-based models directly "out of the box", bypassing critical steps such as algorithm selection and hyperparameter tuning.

In recent years, automated machine learning (AutoML) methods have been introduced to address this Combined Algorithm Selection and Hyperparameter optimization (CASH) problem [13]–[22]. To apply a machine learning-based KAD model, we need a pipeline which involves several components, typically including algorithm selection, data preprocessing, hyperparameter optimization, and network architecture searching. The principal aim of AutoML methods is to automate these facets of the machine learning pipeline, thereby minimizing the human involvement required.

However, to our understanding, the existing AutoML methods addressing the CASH problem necessitate the target algorithm to be supervised. These AutoML techniques require ground truth labels to evaluate the performance of the target algorithm, which influences the subsequent trial of the combination of the algorithm and corresponding hyper-parameters. The absence of dataset labels significantly complicates the performance evaluation process, rendering the current AutoML frameworks unsuitable for practical use.

C. Problem Definition

To the best of our knowledge, there are no AutoML methods designed for KAD to solve the CASH problem in the literature. Our goal is to design an effective and efficient AutoML framework for KPI anomaly detection working on unsupervised anomaly detection algorithms. We define this problem as follows. Given a KPI $\boldsymbol{X} = [(x_1, y_1), \cdots, (x_N, y_N)]$, $x_i \in \mathbf{R}$ and $y_i \in \{0, 1\}$ denote the value at time i and the ground truth of whether it is an anomaly, respectively. Given a set of KAD algorithms $\mathcal{G} = \{g_1, g_2, ..., g_m\}, g_i : x \mapsto y$, and for each algorithm g_i a set of possible hyperparameters $\Theta_i = \{\theta_1^i, \theta_2^i, ..., \theta_{n_i}^i\}$, the goal of CASH for KAD is to find the best g_i and the corresponding configuration θ_j:

$$\arg\max_{g \in \mathcal{G}, h \in \Theta_g} \mathcal{L}(g, \theta; X) \tag{1}$$

$\mathcal{L}(g, \theta; X)$ is a objective function measuring the anomaly detection performance of algorithm g with hyperparameters θ on the KPI X. For easy follow-up, we call a combination of the g and θ a configuration $h = \langle g, \theta \rangle, h \in \mathcal{H}$. So the goal can be equivalently defined as:

$$\arg\max_{h \in \mathcal{H}} \mathcal{L}(h; X) \tag{2}$$

III. Methodology

A. Framework Overview

Fig. 1 illustrates the overall architecture of *AutoKAD*, including three main phases: *cluster-based warm start, **B**ayesian **O**ptimization (BO)-based search, and rule-based configuration recommendation.* The whole workflow is shown in Algorithm 1. We discuss the detail of these three phases in the following sub-sections.

The core module of *AutoKAD* is the BO-based configuration search. Generally, a BO-based AutoML framework has three important components: surrogate model, objective function, and acquisition function. The surrogate model aims to estimate the performance of unobserved configurations according to the observed configurations. **G**aussian **P**rocess (GP) is an efficient and effective surrogate model widely used in BO-based AutoML. Therefore, In *AutoKAD* we adopt GP as our surrogate model.

The role of the objective function is to evaluate the performance of configurations. Thus, the most immediate goal of AutoML is to search for valid configurations to maximize (or minimize) the objective function (lower or bigger). For datasets with ground truth, the final evaluation metric (*e.g.*,

Algorithm 1 The Workflow of *AutoKAD*

Input: $g_\theta^i \in \mathcal{G}, i \in \{1, ..., n\}$: n different unsupervised KAD algorithms hyper-parameterized by θ
Input: X: incoming KPI
Input: $\theta \in \Theta$: Hyperparameter space
Input: f: Our MSE-NF objective function
Input: SW-EI: Our acquisition function
Input: D_h: History tuned KPIs

Phase 1: Clustered-based warm start

apply ROCKA to D_h and get clusters $\mathcal{C} = \{C_1, C_2, ...\}$
assign X to an existing cluster C_i
find the best configurations on cluster centroid KPI X_{C_i} for n KAD algorithms
generate initial configuration set $H_{init} = \{h_1^*, ..., h_n^*\}$

Phase 2: BO-based configuration search

apply f to $H_{init}, f(H_{init}) = \{f(h_1^*), ..., f(h_n^*)\}$ on X
let $H = H_{init}$
while under time and trial limit **do**
  use GP to estimate the $\mu(h), \sigma(h)$ of all possible configurations $\{h_1, ..., h_k\}$ according to H
  apply acquisition function SW-EI to $\{h_1, ..., h_k\} \rightarrow \{\text{SW-EI}(h_1), ..., \text{SW-EI}(h_k)\}$
  find $h_{next} = \arg\max$ SW-EI(h_i)
  H append h_{next}
  train g_θ under the configuration h_{next}, and observe the performance $f(h_{next})$
end while

Phase 3: Rule-based configuration recommendation

manually set the candidate number m
use 3σ rule to remove outliers in H according to MSE on the validation set of X
sort H by the precision (MSE), recall (NF), or F1-score (MSE-NF) preference in ascending order

Output: the first m configurations of sorted H

F1-score, accuracy, or AUC) is usually taken as the objective function. Since high-quality labels are expensive in real-world environments, faced with unlabeled KPI datasets, there is no existing objective function. To solve this problem, we propose the MSE-NF objective function, which, according to experiments, is an excellent approximation of the F1-score.

The acquisition function is used to decide the next configuration to observe and always needs to tackle the exploration-exploitation trade-off [25]. For unsupervised KPI anomaly detection, the gap between F1-score and our objective function could affect the performance of the acquisition function. To address this issue, we propose **S**imilarity **W**eighted **E**xpected **I**mprovement (SW-EI) to enhance the exploration process and alleviate the impact brought by the gap.

B. Clustered-based Warm Start

Generally, an AutoML model needs to train many models with different algorithms and parameters to find the best configuration. This process is usually an exploration-exploitation trade-off. In the beginning, the model tends to select the initial algorithms and hyperparameters randomly due to the lack of prior knowledge about the dataset. However, such a cold start process is unstable and could consume too many computing resources, making the searching process inefficient.

To tackle this problem, we design a warm start mechanism shown in Fig. 3 based on KPI clustering to save search time. The core idea is that the best algorithms and hyperparameters of similar KPIs are likely to be similar. Our inspiration is derived from the process where humans try out initial configurations in real-world environments. Faced with incoming KPI data, human intuition seeks out similar historical KPI instances and attempts to apply algorithms and hyperparameters that were used on these similar KPIs in the past. This approach is not only intuitive, but it has also proven to be highly effective in practice. Therefore, we could leverage the successful experience of similar historical KPIs and adopt their best algorithms and hyperparameters as the initial selection.

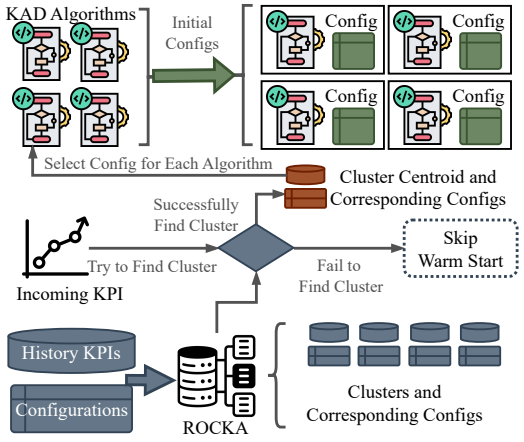


Fig. 3. The workflow of our cluster-based warm start mechanism

Specifically, we first adopt ROCKA [26], a robust and rapid time series clustering algorithm designed for large-scale read-world KPI data, to cluster historical KPIs. For each cluster, ROCKA finds a representative KPI as a centroid for a cluster. Afterward, for a new KPI, ROCKA could assign it to the corresponding cluster to use the best configurations of this cluster centroid for different KAD algorithms as the initial points of BO. In this way, compared with random selection, cluster-based warm start could provide a better starting point, to reduce the search time significantly. The effectiveness of the warm start mechanism will be demonstrated in Section IV-E.

C. Objective Function

The goal of an objective function in the AutoML framework is to evaluate the performance of the algorithms tuned by the AutoML framework, which is a little different from that of the objective functions in ML model training. We will introduce our novel objective function, MSE-NF (Mean Squared Error with Normal Factor), in the following.

After obtaining the initial configuration, *AutoKAD* will start the search process. In general, AutoML models need objective functions to evaluate current performance and determine the next search direction. In the problem of KPI anomaly detection, it is straightforward to define an objective function for labeled data, for example, precision, recall, and F1-score. However, as presented in Section I, it can be a daunting task to obtain high-quality and sufficient labeled KPIs in the real world. Thus, designing the objective function under an unsupervised scenario is a significant challenge.

To tackle this challenge, we design a novel objective function named MSE-NF. Most unsupervised KAD algorithms are based on prediction or reconstruction. Both prediction algorithms and reconstruction algorithms can estimate the normal KPI from the raw KPI data, which is the input of MSE-NF. MSE-NF is a good way to evaluate the quality of prediction or reconstruction. MSE-NF has two parts, namely MSE and NF. MSE can evaluate the similarity between the prediction or reconstruction and the raw KPI. We expect the algorithms to get a low MSE, but the algorithms may overfit if the MSE is too low, making the algorithms unable to find abnormal data points. To solve this problem, we propose the Normal Factor (NF), working like a regularization. NF can evaluate the normal level of the prediction or reconstruction. If the algorithms overfit, the NF will be large because the estimated KPI probably has many anomalies.

MSE is a statistical estimator measuring the distance between the predicted values and the ground truth. Let vector $\boldsymbol{X}$ represent the raw KPI and $\hat{\boldsymbol{X}}$ represent the estimated KPI given by the unsupervised anomaly detection algorithms. N is the number of points of the KPI, x_i is the raw value of the i-th point of the KPI, and the $\hat{x}_i$ is the estimated value given by the unsupervised KPI anomaly detection algorithm. The definition of MSE is shown as follows:

$$\mathit{MSE}(\boldsymbol{X}, \hat{\boldsymbol{X}}) = \sum_{i=1}^{N} \frac{(x_i - \hat{x}_i)^2}{N} \tag{3}$$

In general, a lower MSE indicates that the algorithm can estimate the KPI more accurately. However, if the MSE is too low, the detection algorithm may overfit, making the estimated KPI too similar to the real one. Under this condition, anomalies cannot be found effectively. Therefore, MSE alone is not sufficient for an objective function. If we can evaluate the normality of the estimated KPI given by the KAD algorithm, we may alleviate the overfitting problem. Thus, we propose the Normal Factor. Inspired by the uniformity regularizer in contrasting learning, NF is proposed based on **L**ocal **O**utlier **F**actor (LOF) [23] and can evaluate the normality of KPIs. LOF is a standard method for event detection based on k-nearest neighbors (kNN). Eq. (4) shows the definition of LOF.

$$\mathit{LOF}_k(x) = \frac{1}{|N_k(x)|} \cdot \sum_{x' \in N_k(x)} \frac{\sum_{\hat{x} \in N_k(x)} rd_k(x, \hat{x})}{\sum_{\overline{x} \in N_k(x')} rd_k(x', \overline{x})} \tag{4}$$

rd_k is the reachability-distance of two points x and $\hat{x}$. We use Euclidean distance $d(\cdot)$ to measure the distance in this paper. $N_k(x)$ is the set of x's k-nearest neighbors. To adapt LOF on KPI data, we use time-delay embedding. LOF is a useful method to measure the normality of the data points. Intuitively, we can evaluate the normality of KPIs by evaluating all the normality of KPIs' points. Time-delay embedding is necessary for adapting LOF to KPI data [27], which could transform a time series into a matrix for distance calculating.

In real-world scenarios, the majority of KPIs manifest seasonal trends as a result of user behaviors. Identifying data from the closest corresponding period is not only more efficient but also holds greater significance than merely locating the nearest neighbors. This is attributed to the fact that in the context of time series, a straightforward distance metric between two time points lacks tangible physical meaning. However, data points within adjacent cycles are frequently indicative of potentially similar patterns. Consequently, the NF methodology favors the two periods immediately preceding and following point x, as opposed to the conventional approach of utilizing the k-nearest data points. Therefore, We propose **P**eriod **O**utlier **F**actor (POF) inspired by LOF. The calculation of POF is shown as follows:

$$\begin{aligned} \mathit{POF}(x_t) = & \frac{1}{2} \cdot \frac{d(x_t, x_{t-p}) + d(x_t, x_{t+p})}{d(x_t, x_{t-p}) + d(x_{t-p}, x_{t+p})} \\ & + \frac{d(x_t, x_{t-p}) + d(x_t, x_{t+p})}{d(x_t, x_{t+p}) + d(x_{t-p}, x_{t+p})} \end{aligned} \tag{5}$$

We apply **F**ast **F**ourier **T**ransform (FFT) to estimate the period p of a KPI [28]. If the p given by FFT is not valid, we will set $p = 1$ for further computation of NF. NF is the average POF of all data points embedded by time delay. Eq. (6) shows the definition of NF. In this equation, $\boldsymbol{X}$ is the set of all data points after time-delay embedding. From the equation, we can see that a larger NF indicates fewer normal data points in the KPI, meaning that the estimated KPI may have more anomalies.

$$\mathit{NF}(\boldsymbol{X}) = \frac{1}{|\boldsymbol{X}|} \cdot \sum_{x \in \boldsymbol{X}} \mathit{POF}(x) \tag{6}$$

Usually in BO, we are used to maximizing the objective function, so we use the negative of MSE-NF as our objective function. The effectiveness of the objective function designed by ourselves will be illustrated in Section IV-F.

$$\mathit{MSE\text{-}NF} = \mathit{MSE} + \alpha \mathit{NF} \tag{7}$$

The coefficient $\alpha > 0$ is utilized to balance between MSE and NF. According to our experimental results (see Section IV-F for more details), we opt for $\alpha = 1$.

D. Acquisition Function

In Bayesian optimization (BO), the acquisition function guides the selection of the next query point, balancing the exploration of untested areas and the exploitation of known good ones. It leverages the posterior predictive distribution from the Gaussian Process to estimate the potential improvement at new points, driving the efficient search for global optima.

Expected Improvement (EI) is an effective and widely used acquisition function in BO-based AutoML. Given an unob-

served configuration h and a set of observed configurations $\{h_1, \cdots, h_k\}$, the EI function can be written as Eq. (8).

$$EI(h) = \mathbb{E}[\max(\mu(h) - f(h^+), 0) \mid \{h_1, \cdots, h_k\}] \quad (8)$$
$$\text{where} \quad h^+ = \arg\max_{i=1,\cdots,k} f(h_i)$$

$\mu(h)$ is an estimated value of the unobserved configuration h given by the surrogate model in BO. If we use Gaussian Process (GP) as the surrogate model in BO, then the EI function can be rewritten as follows.

$$EI(h) = \begin{cases} \left(\mu(h) - f\left(h^+\right)\right)\Phi(z) + \sigma(h)\phi(z) & \sigma > 0 \\ 0 & \sigma = 0 \end{cases} \quad (9)$$
$$\text{where} \quad z = \frac{\mu(h) - f\left(h^+\right)}{\sigma(h)}$$

$\Phi(\cdot)$ and $\phi(\cdot)$ are the cumulative distribution function and probability density function of the Gaussian distribution, respectively. $\sigma(h)$ denotes the standard deviation estimated by the GP.

Although our proposed MSE-NF objective function is an effective label-free way to evaluate the performance of KAD algorithms. However, there is still a discrepancy between MSE-NF and the real F1-score. The commonly used EI acquisition function in the AutoML framework always uses the minimum (or maximum) value of the observed points given by the objective function as an essential measure. The discrepancy between the objective function and the true evaluation metric (i.e., F1 score) may affect the accuracy of the EI function. Since we have no labels, the gap between F1 and our objective function cannot be completely eliminated, so an intuitive solution is to be more inclined to explore different configurations rather than obsessing over finding the maximal value of the objective function.

To address this issue, we propose the **S**imilarity **W**eighted **E**xpected **I**mprovement (SW-EI). The "similarity" in SW-EI denotes the similarity between two different configurations. Since different KAD algorithms may have different configuration spaces, it is infeasible to calculate the similarity between two configurations from different algorithms. Therefore, we define that the similarity of two different algorithms' configurations is 0. Then, we normalize the hyperparameters in the configuration space to $[0, 1]$ for each KAD algorithm. For the configurations from the same algorithm, we use cosine similarity:

$$\text{simi}(h_1, h_2) = \frac{h_1 \cdot h_2}{|h_1| \times |h_2|} \quad (10)$$

Given a set of observed configurations $\{h_1, \cdots, h_k\}$, let n stand for the number of different algorithms in configuration spaces. The SW-EI is defined as follows:

$$SW\text{-}EI(h) = \sum_{i=1}^{n} w_i(h, h_i^+)(\mu(h) - f(h_i^+))\Phi(z_i) \quad (11)$$
$$+ \sum_{i=1}^{n} w_i \sigma(h)\phi(z_i) + \sigma(h)\phi(\sum_{i=1}^{n} \frac{z_i}{n})$$
$$\text{where} \quad z_i = \frac{\mu(h) - f\left(h_i^+\right)}{\sigma(h)}, \quad w_i = \frac{\text{simi}(h, h_i^+)}{\sum_{j=1}^{n} \text{simi}(h, h_j^+)}$$

h_i^+ denotes the best configuration for the i-th algorithm. The core design concept of SW-EI is to drive the model to explore configurations that are different from those already observed while guaranteeing the expected improvement. SW-EI considers the best configuration of different algorithms and the similarity of configurations, making the AutoML model tend to explore the potential of different algorithms and prevent the model from limiting itself to a particular algorithm or similar hyperparameters.

E. Configuration Recommendation

The goal of configuration recommendation is to recommend different configurations according to operators' preferences. For labeled data, the operators can utilize some evaluation metrics like precision, recall, and F1-score to evaluate the configurations and select satisfying ones. However, it is challenging for operators to pick satisfying configurations without evaluation metrics based on labels. Besides, operators may have different preferences regarding precision and recall for different KPIs. For some KPIs, the operators expect a high precision of the anomaly detection algorithm because the anomalies of these KPIs are not crucial. Too many false alarms influence the efficiency of the operation work. However, for some crucial KPIs, the operators expect a high recall of the anomaly detection algorithm because any anomaly in these KPIs may lead to significant revenue loss, thus the anomaly detection algorithm must report any potential anomaly.

We propose a rule-based strategy of configuration recommendation to solve the above problems. The utility of MSE lies in its ability to quantify the degree of similarity between the original KPI and its estimated counterpart produced by the unsupervised detector. A detection algorithm that results in a low MSE value suggests that it can accurately estimate the KPI, thus enabling it to detect only significant anomalies, leading to enhanced precision. On the other hand, NF serves as a measure of the normality of the estimated KPI. When an algorithm delivers a low NF, it suggests effective learning of the normal patterns within the KPI. Consequently, the estimated KPI tends to mirror a normal KPI as closely as possible, facilitating the detection of even minor anomalies and thereby contributing to elevated recall. MSE-NF balances the precision and the recall leading to a high F1-score. So MSE, NF, and MSE-NF are good estimates for precision, recall, and F1-score, respectively.

More specifically, our strategy first averages the MSE and NF of all the configurations for the KPI. Subsequently, we deploy the 3σ rule to discard any outlier configurations that fall beyond three standard deviations from the calculated means. The selection of the 3σ rule is motivated by our experimental findings, wherein it is observed that the outlier configurations typically exhibit extraordinarily high precision or recall. However, significantly high precision or recall usually leads to a notably low F1-score. After removing the outlier

configuration, we sort the observed configurations in ascending order. Then, our strategy selects top m configurations, and the operators can set the m by themselves. The superiority of our recommendation strategy will be presented in Section IV-G.

IV. Evaluation

In this section, we evaluate *AutoKAD* using various real-world KPIs, aiming to answer the following research questions:

- RQ1: How effective is *AutoKAD* in searching for suitable configurations?
- RQ2: How much can the warm start mechanism accelerate the searching process?
- RQ3: Is the MSE-NF objective function effective in estimating the performance of the unsupervised KPI anomaly detection algorithms?
- RQ4: Can the strategy of configuration recommendation correctly recommend configurations?

A. Dataset

To demonstrate the effectiveness of *AutoKAD*, we conduct experiments on three datasets. The first dataset is the 2018 international AIOps algorithm competition dataset, a public dataset [29]. The second and third datasets are collected from two core trading systems deployed in a large international commercial bank.

In dataset $\mathcal{A}$, there are 29 different KPIs collected from big Internet companies. These KPI data are from the real-world production environment and labeled by domain experts. Therefore, these labeled data can effectively reflect the performance of our approach in a real-world environment. Since these KPIs have a diverse range of patterns and physical meanings, this dataset has been used to evaluate the robustness of KPI anomaly detection in many advanced work [30]–[33].

Dataset $\mathcal{B}$ and $\mathcal{C}$ contain 29 and 30 KPIs, respectively, collected from the two most important trading systems of an international commercial bank. There are dozens of operators closely monitoring these KPIs to maintain service quality. All the KPIs in these datasets are 3 months long at the minute granularity and labeled by experienced operators.

Each KPI is divided into the training set and the test set. The length and distribution of each KPI's test set are similar to the training set's. Table I presents some statistical information about three datasets, such as the number of KPIs, the number of data points, and the anomaly rate. Though our approach focuses on hyperparameter optimization of unsupervised KPI anomaly detection algorithms, we still need labels for the purpose of evaluation. The ground truth of all anomalies in the dataset is only used to evaluate the performance of our hyperparameter optimization model. Neither the training nor testing phase of anomaly detection algorithms uses labels.

TABLE I
Dataset statistic

Dataset	#KPI	#Train/#Test	Anomaly rate
$\mathcal{A}$	29	3004066 / 2918847	2.648% / 1.869%
$\mathcal{B}$	29	1642815 / 1642810	1.089% / 1.065%
$\mathcal{C}$	30	2078848 / 2078854	0.781% / 0.817%

B. Candidate Algorithms and Hyperparameter Space

We choose three representative unsupervised KPI anomaly detection algorithms in our experiments, including a traditional method and typical machine learning methods for *AutoKAD*. Table II shows the space of algorithms and hyperparameters.

The Holt-Winters based on exponential smoothing is a classic and successful method for unsupervised KPI anomaly detection [5]. It is practical and efficient for seasonal KPIs with obvious anomaly patterns. Besides, Holt-Winters has good interpretability, making it popular in the industry. LSTM-NDT [8] is a typical prediction-based unsupervised KPI anomaly detection algorithm. LSTM-NDT can capture the long-term dependency and information for the KPI data. Donut is a representative reconstruction-based unsupervised KPI anomaly detection algorithm designed for seasonal KPIs [7]. Due to the solid theoretical explanation of Donut, it is popular in the industry and literature.

TABLE II
Algorithms and their different types of hyperparameters included in the experiments of *AutoKAD*.

Algorithm	Mechanism	Hyperparameter	Type
Holt-winters [5]	Pred	trend	category
		seasonal	category
		damped rend	boolean
		seasonal periods	integer
		init method	category
LSTM-NDT [8]	Pred	window length	integer
		batch size	integer
		learning rate	float
		epoch number	integer
		hidden dimension	integer
Donut [7]	Recons	window length	integer
		batch size	integer
		learning rate	float
		epoch number	integer
		hidden dimension	integer
		latent dimension	integer

C. Evaluation Metrics

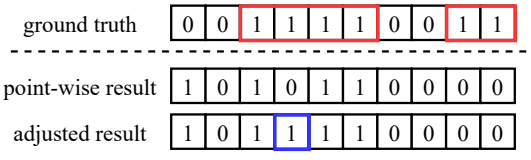


Fig. 4. An illustration of the adjustment strategy employed in the evaluation metrics. Anomaly points within the ground truth are depicted as red rectangles, while the points that have been adjusted are signified by blue rectangles.

TABLE III
PRECISION, RECALL, AND F1-SCORE COMPARISON BETWEEN *AutoKAD* AND THREE BASELINE METHODS UNDER THE LIMITATION OF 1 HOUR SEARCHING TIME AND MAX ITERATION OF 100.

Methods	$\mathcal{A}$			$\mathcal{B}$			$\mathcal{C}$			$Avg.$		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
Default Donut	0.881	0.380	0.531	0.457	0.689	0.550	0.707	0.638	0.671	0.682	0.569	0.620
Random Search	0.676	0.537	0.599	0.663	0.631	0.647	0.675	0.783	0.725	0.671	0.650	0.661
BayesOpt	0.876	0.525	0.657	0.763	0.672	0.715	0.875	0.681	0.766	0.838	0.626	0.717
BOAT	0.831	0.575	0.680	0.778	0.688	0.730	0.823	0.708	0.761	0.811	0.657	0.726
AutoKAD	0.861	0.694	**0.769**	0.920	0.723	**0.810**	0.916	0.781	**0.843**	0.899	0.733	**0.807**

Precision, recall, and F1-score are common metrics to evaluate the performance of anomaly detection [34]. However, in the KAD area, the operators in the industry typically disregard point-wise metrics. Since, we adopt an adjustment precision, recall, and F1-score widely used in previous KAD works [7], [10], [11], [30], [35], [36]. In the adjustment metrics, if any point in an anomaly interval in the ground truth can be detected, we consider this interval to be detected correctly, and all points in this interval are treated as they can be detected. The adjustment process is illustrated in Fig. 4.

D. RQ1: Overall Performance of AutoKAD

In order to demonstrate the effectiveness of *AutoKAD*, we compare it with four baseline methods.

- **Default configuration**. We use a popular unsupervised KPI anomaly detection algorithm (*i.e.*, Donut [7]) as the default algorithm, and adopt the default hyperparameters used in the open-source project [1].
- **Random search**. Random search can randomly search the space of combinations of algorithms and hyperparameters and return one configuration in every iteration.
- **Bayesian optimization (BayesOpt)** [21]. In this experiment, BayesOpt uses the MSE as the objective function and the original EI as the acquisition function.
- **BOAT** [37]. The state-of-the-art auto-tuner is based on structured Bayesian optimization. Like BayesOpt, BOAT uses the MSE as the objective function.

In this experiment, we limit the max iteration to 100 and the total searching time (including training machine learning models) to 1 hour for each KPI. We record the best F1-score of configurations returned by the models. It is noteworthy that since BayesOpt and BOAT are designed for labeled datasets, they cannot natively handle the CASH problem for unsupervised KAD. MSE is usually the loss function of many KAD methods [1], [2], [4], [8], [12], [35]. So we let BayesOpt and BOAT use MSE as the objective function. The warm start mechanism is specifically designed for *AutoKAD*, so it can not easily adapt to other baseline methods in the experiments. Therefore, to ensure the fairness of the evaluation, we do not activate the warm start mechanism. Table III shows the overall performance of *AutoKAD* and three baseline methods.

From Table III, we can see *AutoKAD* exhibits outstanding performance on all three datasets and achieve the F1-score of 0.807 on average. We further analyze the weakness of compared methods and the superiority of *AutoKAD* in depth. For default configuration, Donut is specifically designed for seasonal KPIs, leading to an unsatisfying performance on stable and unstable KPIs. Besides, due to the variety of KPIs, it is unrealistic for an algorithm to perform well on all data with the same hyperparameters. Thus, the overall performance of Donut is far from satisfying. In terms of random search, as the most straightforward method for hyperparameter tuning, it can not capture the relationship between the configurations and performance. Therefore, it can not focus on suitable configurations, wasting time on inferior configurations.

In terms of the effectiveness of BayesOpt and BOAT, they are deemed less efficient when compared with *AutoKAD*. We believe that the reasons behind this discrepancy originate from two aspects. Firstly, the fundamental EI metric overemphasizes improving the maximum reward for the currently observed points, while neglecting to explore varying KAD configurations. In contrast, SW-EI not only focuses on enhancing the current optimal solution but also encourages the exploration of distinct configurations, thereby effectively preventing falling into local optimum solutions. Secondly, both BayesOpt and BOAT employ MSE as their objective function. However, MSE does not serve as an effective performance evaluation measure for KAD. This shortcoming hampers both methods' capability in identifying the algorithm and hyperparameter combinations that provide superior performance in KAD. In contrast, *AutoKAD*'s MSE-NF proves to be a more effective measure for evaluating KAD performance. A more detailed analysis can be found in Section IV-F.

In summary, compared with baseline methods, the results show that *AutoKAD* is indeed effective in the CASH problem for unsupervised KAD models.

E. RQ2: Contribution of Warm Start to Time Efficiency

The warm start mechanism can reduce the time spent on exploration at the beginning. To evaluate the performance of our warm start mechanism, we divide all the KPIs in the three datasets into two parts: one part is considered as a tuned part, and the other part is to be tuned. *AutoKAD* has already tuned the KPIs in the tuned part. We run ROCKA on this part, obtaining four clusters. The other part includes 9 KPIs named by numbers from 1 to 9, evenly from the three datasets.

To evaluate the performance of our warm start mechanism, we run *AutoKAD* twice on these 9 KPIs with the limitation

[1]https://github.com/NetManAIOps/donut

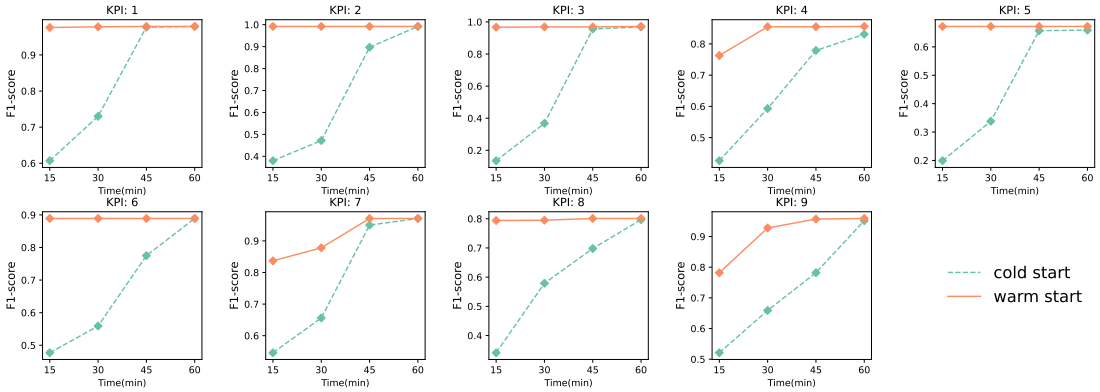


Fig. 5. The best F1-score of *AutoKAD* under cold start and warm start on 9 KPIs at 15min, 30min, 45min, and 60min.

of 1 hour running time and max iteration of 100. The first time we do not activate the warm start mechanism suffering from the cold start. Moreover, the second time, we activate the warm start replacing 5 initial points of *AutoKAD*. Fig. 5 shows the best F1-score of configurations at 15min, 30min, 45min, and 60min.

The results show that *AutoKAD* equipped with our warm start mechanism can focus on good configurations much faster than cold start. At 15min, the best F1-score of warm start is 164% higher than the best F1-score of cold start on average. At 60min, the best F1-scores of all the KPIs with warm start are no less than the cold start's, indicating that the warm start will not reduce the overall performance. Besides, for KPI 2, 5, and 6, the *AutoKAD* has already converged at 15min indicating that the model thinks the best configuration has been found. For most KPIs in this RQ, the best F1-scores of cold start at 45min are close to warm start's at 15min. It indicates that our warm start can save about 30min with the limitation of 1 hour running time. The results indicate that our warm start mechanism can effectively learn successful experiences from previous similar KPIs. With great initial configurations, *AutoKAD* can accurately focus on the configurations deserving investigation, reducing the search space.

F. RQ3: Effectiveness of the Objective Function

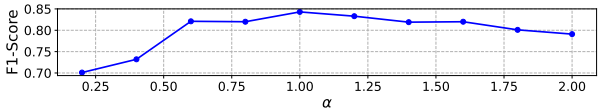


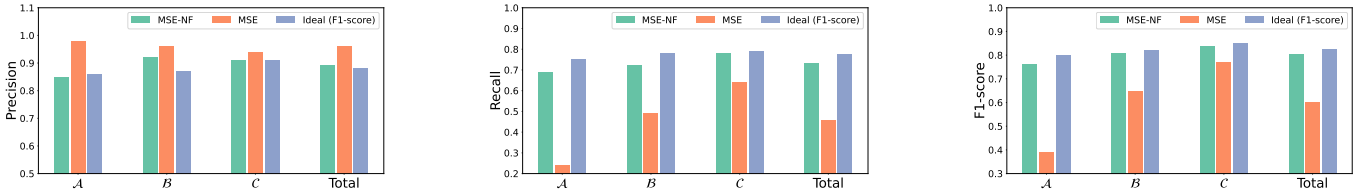
Fig. 6. The sensitivity analysis of the parameter, α in MSE-NF

MSE-NF objective function is an estimator of F1-score of KPI anomaly detection. In the real world, most KPIs are unlabeled, and F1-scores can not be computed without labels. For evaluation purposes, we use labels to compute the F1-score as the ideal objective function to acquire the optimal configurations. To demonstrate the validation of MSE-NF, we compare the MSE-NF's performance with the best F1-score and MSE. In this RQ, we use *AutoKAD* as the AutoML framework and choose different objective functions. The ideal group uses the best F1-score's performance straightforwardly as the objective function, and the MSE group uses MSE as the objective function. All the groups have the limitation of 1 hour of searching time and a max iteration of 100. Fig. 7 shows the performance comparison of these three objective functions.

From the results, we can see that the performance of MSE-NF is quite close to the performance of the ideal, demonstrating that MSE-NF is a suitable estimator of F1-score. MSE can achieve the best precision compared with the other objective functions, but the recall and F1 are much lower. MSE aims to evaluate the similarity between the raw KPI and the estimated KPI given by the unsupervised detector. However, too low an MSE may indicate that the detection algorithm overfits and exactly predicts or reconstructs the anomaly points in the test data. In this situation, the detection algorithm learns not only the normal pattern but also the abnormal pattern from the training data, which fails to observe the original intention of unsupervised KPI anomaly detection algorithms. Therefore, only the extreme anomaly points are recognized as anomalies making many anomalies undetected. So MSE objective function can achieve pretty high precision (Fig. 7a), but the F1-score (Fig. 7c) is not satisfying. In the MSE-NF function, NF works like regularization. It can prevent the detection algorithm from overfitting to anomalies. We use Dataset $\mathcal{C}$ which most accurately reflects the real-world environment to conduct a sensitivity analysis of the paramter α in MSE-NF and the result is shown in Fig. 6. The result demonstrates that when α is close to 1, the performance is commendable. Therefore, in our method, we choose $\alpha = 1$.

G. RQ4: Correctness of the Configuration Recommendation

Our strategy of configuration recommendation can recommend configurations according to operators' preferences. We run *AutoKAD* with the same limitation in RQ1 and our strategy recommended top 3 configurations. We compare the configurations given by the strategy with the optimal configurations to validate whether our strategy can recommend close-to-optimal and even optimal configurations. It is noteworthy that in this RQ, our strategy recommends the best precision configuration, best recall configuration, and best F1-score configuration sep-



(a) Precision (b) Recall (c) F1-score

Fig. 7. Performance comparison of MSE-NF, MSE, and F1-score.

arately. Therefore, even for the same KPI, the best precision configuration and the best recall configuration can be from different algorithms and hyperparameters. Table IV shows the comparison result.

The results show that our strategy can correctly recommend configurations according to different precision, recall, and F1-score preferences. For high precision preference, the most significant difference between our strategy and optimal configurations is at most 0.02. For high F1-score preference, the difference is at most 0.05. For high recall preference, the difference is at most 0.11. The recommended configurations of high recall preference may differ a little from the optimal configurations. It is because the NF guiding recommendation of high recall preference is a regularization term computed only based on the estimated KPI. Therefore NF can not capture the relationship between the estimated KPI and raw KPI. In our practice, we find that a low NF with a low recall is usually an outlier of the configurations. So we use the 3σ rule to eliminate the outlier, improving the performance of the high recall preference recommendation.

TABLE IV
TOP 3 RECOMMENDED CONFIGURATIONS COMPARISON BETWEEN OUR STRATEGY AND OPTIMAL CONFIGURATIONS

Top 3	Dataset	$\mathcal{A}$			$\mathcal{B}$			$\mathcal{C}$		
	Strategy	P	R	F1	P	R	F1	P	R	F1
1st	ideal	0.99	0.88	0.76	0.99	0.96	0.81	0.97	0.87	0.84
	our	0.97	0.77	0.71	0.98	0.95	0.78	0.97	0.81	0.84
2nd	ideal	0.99	0.77	0.73	0.98	0.95	0.80	0.97	0.83	0.83
	our	0.98	0.71	0.71	0.97	0.88	0.78	0.97	0.81	0.83
3rd	ideal	0.99	0.71	0.70	0.98	0.92	0.78	0.97	0.82	0.83
	our	0.98	0.70	0.66	0.97	0.89	0.78	0.96	0.80	0.82

V. Related Work

KPI anomaly detection is crucial for large-scale online service systems, and many efforts have been devoted to this field [1], [2], [5]–[9], [24], [38]–[40]. In practice, model selection and hyperparameter tuning have a significant influence on the performance of algorithms. However, it is challenging for human operators lacking expert knowledge about algorithms to select a proper model and tune the hyperparameters. Some ensemble models such as [1], [2] aim to tackle this problem. These models ensemble various supervised KPI anomaly detection algorithms and use a machine learning model like a random forest to select appropriate detectors and hyperparameters. However, these ensemble models need labels of KPI, but most KPIs in practice are unlabeled because of the enormous number of KPIs and the time cost. Therefore, there is a great limitation of these ensemble models in practice.

A more general way to solve CASH problem for KAD is AutoML. Many AutoML approaches are proposed to solve the hyperparameter optimization problem [13]–[22], [37]. These methods can capture the relationship between the performance and the hyperparameter settings. Therefore, the methods can pay more attention to the hyperparameter settings deserving investigation, reducing the time for searching for promising settings. However, all these methods need labels, and none of them is natively designed for unsupervised KPI anomaly detection.

VI. Conclusion

KPI anomaly detection is critical for service quality and user experience in large-scale online service systems. Due to the overwhelming number of KPIs and the complexity of detection algorithms, it is challenging for human operators to select a promising algorithm and tune the hyperparameters for each KPI. In this paper, we propose an automatic algorithm selection and hyperparameter tuning framework called *AutoKAD* for KPI anomaly detection. *AutoKAD* can effectively and efficiently find a good configuration for a given KPI. Using our carefully designed objective and acquisition function, *AutoKAD* can tune unsupervised KPI anomaly detection algorithms, which is a challenge for existing AutoML frameworks. The warm start mechanism can significantly reduce the time required for AutoML cold start. In addition, the configuration recommendation strategy can accurately recommend configurations according to operator preferences. We conduct experiments on real-world data to evaluate the performance of *AutoKAD*. The experiments show that *AutoKAD* is effective and achieves an F1-score of 0.807 within 1 hour for each KPI on our dataset. The F1-score of *AutoKAD* is up to 40% higher than a widely used Donut model. Our cluster-based warm start mechanism can help *AutoKAD* achieve a competitive F1-score within 15 minutes compared to a 1 hour cold start search.

Acknowledgment

This work is supported by the National Key Research and Development Program of China (No.2019YFE0105500), the Research Council of Norway (No.309494), the National Natural Science Foundation of China (Grant No.62072264 No.62202445), and the Beijing National Research Center for Information Science and Technology (BNRist) key projects.

REFERENCES

[1] D. Liu, Y. Zhao, H. Xu, Y. Sun, D. Pei, J. Luo, X. Jing, and M. Feng, "Opprentice: Towards practical and automatic anomaly detection through machine learning," in *Proceedings of the 2015 Internet Measurement Conference*, 2015, pp. 211–224.

[2] N. Laptev, S. Amizadeh, and I. Flint, "Generic and scalable framework for automated time-series anomaly detection," in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2015, pp. 1939–1947.

[3] J. Bu, Y. Liu, S. Zhang, W. Meng, Q. Liu, X. Zhu, and D. Pei, "Rapid deployment of anomaly detection models for large number of emerging KPI streams," in *37th IEEE International Performance Computing and Communications Conference, IPCCC 2018, Orlando, FL, USA, November 17-19, 2018*. IEEE, 2018, pp. 1–8.

[4] X. Zhang, Q. Lin, Y. Xu, S. Qin, H. Zhang, B. Qiao, Y. Dang, X. Yang, Q. Cheng, M. Chintalapati, Y. Wu, K. Hsieh, K. Sui, X. Meng, Y. Xu, W. Zhang, F. Shen, and D. Zhang, "Cross-dataset time series anomaly detection for cloud systems," in *2019 USENIX Annual Technical Conference, USENIX ATC 2019, Renton, WA, USA, July 10-12, 2019*, D. Malkhi and D. Tsafrir, Eds. USENIX Association, 2019, pp. 1063–1076.

[5] C. Chatfield and M. Yar, "Holt-winters forecasting: some practical issues," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 37, no. 2, pp. 129–140, 1988.

[6] H. Zare Moayedi and M. Masnadi-Shirazi, "Arima model for network traffic prediction and anomaly detection," in *2008 International Symposium on Information Technology*, vol. 4, 2008, pp. 1–6.

[7] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng *et al.*, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 187–196.

[8] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Söderström, "Detecting spacecraft anomalies using lstms and non-parametric dynamic thresholding," in *KDD*. ACM, 2018, pp. 387–395.

[9] W. Chen, H. Xu, Z. Li, D. Pei, J. Chen, H. Qiao, Y. Feng, and Z. Wang, "Unsupervised anomaly detection for intricate kpis via adversarial training of vae," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 1891–1899.

[10] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *KDD*. ACM, 2019, pp. 2828–2837.

[11] Z. Li, Y. Zhao, J. Han, Y. Su, R. Jiao, X. Wen, and D. Pei, "Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding," in *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, F. Zhu, B. C. Ooi, and C. Miao, Eds. ACM, 2021, pp. 3220–3230.

[12] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly transformer: Time series anomaly detection with association discrepancy," in *ICLR*. OpenReview.net, 2022.

[13] M. Feurer, A. Klein, J. Eggensperger, Katharina Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Advances in Neural Information Processing Systems 28 (2015)*, 2015, pp. 2962–2970.

[14] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6765–6816, 2017.

[15] E. Hazan, A. Klivans, and Y. Yuan, "Hyperparameter optimization: A spectral approach," *arXiv preprint arXiv:1706.00764*, 2017.

[16] V. Dalibard, M. Schaarschmidt, and E. Yoneki, "Boat: Building auto-tuners with structured bayesian optimization," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 479–488.

[17] O. Alipourfard, H. H. Liu, J. Chen, S. Venkataraman, M. Yu, and M. Zhang, "Cherrypick: Adaptively unearthing the best cloud configurations for big data analytics," in *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17)*, 2017, pp. 469–482.

[18] T. Kathuria, A. Deshpande, and P. Kohli, "Batched gaussian process bandit optimization via determinantal point processes," *Advances in Neural Information Processing Systems*, vol. 29, pp. 4206–4214, 2016.

[19] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Auto-weka: combined selection and hyperparameter optimization of classification algorithms," in *KDD*. ACM, 2013, pp. 847–855.

[20] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *ICML (1)*, ser. JMLR Workshop and Conference Proceedings, vol. 28. JMLR.org, 2013, pp. 115–123.

[21] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *NIPS*, 2012, pp. 2960–2968.

[22] C. Freeman, I. Beaver, and A. Mueen, "Improving univariate time series anomaly detection through automatic algorithm selection and human-in-the-loop false-positive removement," in *The International FLAIRS Conference Proceedings*, vol. 34, no. 1, 2021.

[23] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.

[24] H. Yan, A. Flavel, Z. Ge, A. Gerber, D. Massey, C. Papadopoulos, H. Shah, and J. Yates, "Argus: End-to-end service anomaly detection and localization from an isp's point of view," in *2012 Proceedings IEEE INFOCOM*. IEEE, 2012, pp. 2756–2760.

[25] R. Bardenet, M. Brendel, B. Kégl, and M. Sebag, "Collaborative hyperparameter tuning," in *ICML (2)*, ser. JMLR Workshop and Conference Proceedings, vol. 28. JMLR.org, 2013, pp. 199–207.

[26] Z. Li, Y. Zhao, R. Liu, and D. Pei, "Robust and rapid clustering of kpis for large-scale anomaly detection," in *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*. IEEE, 2018, pp. 1–10.

[27] S. Oehmcke, O. Zielinski, and O. Kramer, "Event detection in marine time series data," in *Joint German/Austrian Conference on Artificial Intelligence (Künstliche Intelligenz)*. Springer, 2015, pp. 279–286.

[28] W. W. Wei, "Time series analysis," in *The Oxford Handbook of Quantitative Methods in Psychology: Vol. 2*, 2006.

[29] Z. Li, N. Zhao, S. Zhang, Y. Sun, P. Chen, X. Wen, M. Ma, and D. Pei, "Constructing large-scale real-world benchmark datasets for aiops," *arXiv preprint arXiv:2208.03938*, 2022.

[30] H. Ren, B. Xu, Y. Wang, C. Yi, C. Huang, X. Kou, T. Xing, M. Yang, J. Tong, and Q. Zhang, "Time-series anomaly detection service at microsoft," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 3009–3017.

[31] T. Wu and J. Ortiz, "RLAD: time series anomaly detection through reinforcement learning and active learning," *CoRR*, vol. abs/2104.00543, 2021. [Online]. Available: https://arxiv.org/abs/2104.00543

[32] C. Wang, K. Wu, T. Zhou, G. Yu, and Z. Cai, "Tsagen: Synthetic time series generation for kpi anomaly detection," *IEEE Transactions on Network and Service Management*, 2021.

[33] J. Li, S. Di, Y. Shen, and L. Chen, "Fluxev: a fast and effective unsupervised framework for time-series anomaly detection," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021, pp. 824–832.

[34] N. Tatbul, T. J. Lee, S. Zdonik, M. Alam, and J. Gottschlich, "Precision and recall for time series," in *NeurIPS*, 2018, pp. 1924–1934.

[35] J. Li, S. Di, Y. Shen, and L. Chen, "Fluxev: A fast and effective unsupervised framework for time-series anomaly detection," in *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*, L. Lewin-Eytan, D. Carmel, E. Yom-Tov, E. Agichtein, and E. Gabrilovich, Eds. ACM, 2021, pp. 824–832.

[36] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "USAD: unsupervised anomaly detection on multivariate time series," in *KDD*. ACM, 2020, pp. 3395–3404.

[37] V. Dalibard, M. Schaarschmidt, and E. Yoneki, "Boat: Building auto-tuners with structured bayesian optimization," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 479–488.

[38] J. Chen, N. Xu, P. Chen, and H. Zhang, "Efficient compiler autotuning via bayesian optimization," in *ICSE*. IEEE, 2021, pp. 1198–1209.

[39] C. Wu, N. Zhao, L. Wang, X. Yang, S. Li, M. Zhang, X. Jin, X. Wen, X. Nie, W. Zhang, K. Sui, and D. Pei, "Identifying root-cause metrics for incident diagnosis in online service systems," in *ISSRE*. IEEE, 2021, pp. 91–102.

[40] X. Wang, K. Yin, Q. Ouyang, X. Wen, S. Zhang, W. Zhang, L. Cao, J. Han, X. Jin, and D. Pei, "Identifying erroneous software changes through self-supervised contrastive learning on time series data," in *ISSRE*. IEEE, 2022, pp. 366–377.