# Efficient Multivariate Time Series Anomaly Detection Through Transfer Learning for Large-Scale Web Services

Yongqian Sun†, Minghan Liang†, Zeyu Che†, Dongwen Li†, Tinghua Zheng†,
Shenglin Zhang*†§¶, Pengtian Zhu†, Yuzhi Zhang†§¶, Dan Pei‡
† Nankai University, {sunyongqian, zhangsl, zyz}@nankai.edu.cn,
{minghanliang, czy, lidongwen, tinghuazheng, nkpt}@mail.nankai.edu.cn
‡ Tsinghua University, BNRist, peidan@tsinghua.edu.cn
§ Tianjin Key Laboratory of Operating System (TKL-OS)
¶ Haihe Laboratory of Information Technology Application Innovation (HL-IT)

***Abstract*—Timely anomaly detection of multivariate time series (MTS) is of vital importance for managing large-scale Web services. However, many deep learning-based MTS anomaly detection models require long-term MTS training data to achieve good performance, which conflicts with frequent pattern changes in Web services entities. Moreover, the training overhead of vast MTS in large-scale Web services is unacceptable. To address these issues, we design *OmniTransfer*, a model-agnostic framework that combines improved hierarchical agglomerative clustering with an adaptive transfer learning strategy, making many state-of-the-art (SOTA) MTS anomaly detection models efficient and effective. Extensive experiments using real-world data from a large Web content service provider show that *OmniTransfer* significantly reduces the model initialization time by 59.72% and the training cost by 85.01%, while maintaining high accuracy in detecting anomalies.**

***Index Terms*—Transfer Learning, Multivariate Time Series, Multivariate Time Series Clustering, Anomaly Detection, Phase Shift**

## I. Introduction

With the rapid development of the Internet, the scale of Web services has grown exponentially. Anomaly detection is critical to the quality of service (QoS) management since it helps operators identify anomalous behaviors, improve system stability, and reduce economic losses [1]–[4]. Operators configure multiple monitoring metrics for each entity to monitor the running status, usually collected continuously at predefined intervals. As shown in Fig. 1, the monitored metrics of an entity form a multivariate time series (MTS), including system metrics (e.g., CPU load, memory usage, network throughput and disk I/O) and user-perceived metrics (e.g., average response latency, page visits and access error rates).

Recently, a series of deep learning-based MTS anomaly detection models have been proposed [5]–[10], but they suffer from some limitations. First, they need a long initialization time [1] to perform well. For instance, OmniAnomaly [5] and InterFusion [7] require several weeks of training data. However, operators want to reduce the initialization time when there is a pattern change, such as configuration upgrades or adding new entities. Second, training a model for each entity is impractical as large-scale Web services have massive entities. For example, it takes 99.87 days to 15.40 years to train models for one million entities [12]. Third, models are generally designed for specific purposes, making it difficult to select an optimal algorithm for different scenarios. For example, GDN [10] focuses on the correlation between indicators, while InterFusion [7] additionally considers temporal dependencies. Therefore, a framework that can effectively reduce initialization time and training overhead for all models is needed.

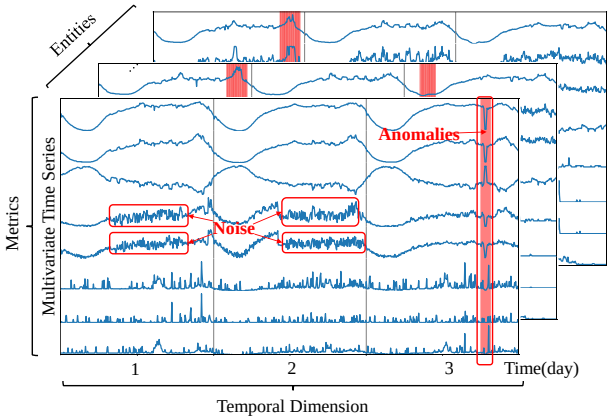


Fig. 1: The MTS of entities in large-scale Web services.

Intuitively, combining clustering with transfer learning is a promising approach to solve this problem [13]. The training overhead is reduced by reducing the number of models that require training through clustering. Then, by quickly fine-tuning the pre-trained model to a new pattern to reduce the initialization time. Only a small amount of data is needed to determine the category and parameter transfer. Note that, we denote the MTS and models in the source domain as

* Shenglin Zhang is the corresponding author (zhangsl@nankai.edu.cn).

[1]MTS's model initialization time [11] is defined as the data time interval between the model startup and the model training.

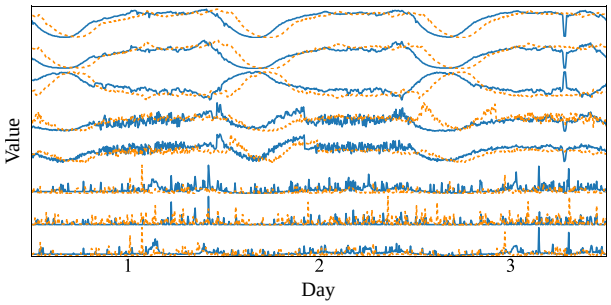


Fig. 2: An example of MTS phase shifts.

the base MTS and base models, respectively, and the MTS and models in the target domain as the target MTS and target models. However, there are still some challenges when applying clustering and transfer learning.

(1) **High diversity of MTS.** The diversity of MTS includes patterns, irregular noise and anomalies. MTS can be generated by various entities with diverse patterns (i.e., different amplitude, trend, etc.). As Fig. 1 shows, MTS contains irregular noises, anomalies, and extreme values (outliers that deviate several times from normal). Fig. 2 illustrates some MTS may have similar shapes but with phase shifts. This can happen when large-scale Web services use different servers to serve users across a wide geographical area, resulting in similar MTS patterns with a time delay. The diversity can affect the distance calculation of MTS and lead to poor clustering performance.

(2) **Selection of transfer strategy.** There are various strategies for transferring parameters from the base model to the target model. Full parameter transfer and partial parameter transfer strategy are two typical strategies. In most cases, we have the following observations: (a) The distances between the base and target MTS are various, making the optimal transfer strategy of each target MTS can be different. (b) The optimal transfer strategies for different models are diverse. Therefore, we need to use adaptive transfer strategies to achieve better detection performance.

To solve the aforementioned issues, we propose *OmniTransfer*, an efficient, unsupervised, and model-agnostic framework for MTS anomaly detection. In the offline training stage, *OmniTransfer* uses an improved hierarchical agglomerative clustering (I-HAC) method to cluster the data. Then, *OmniTransfer* trains a base model for each cluster. When applying the model to a new pattern MTS, *OmniTransfer* assigns the new pattern MTS to the nearest cluster and fine-tunes the base model by an adaptive transfer strategy.

The main contributions of our work are as follows:

(1) To reduce the initialization time and the training overhead for large-scale Web services, *OmniTransfer* uses clustering and transfer learning techniques to transfer the knowledge from well-trained base models to target models.

(2) We propose an adaptive transfer strategy. It can automatically select either full parameter transfer or partial parameter transfer strategy according to the distance between the target MTS and the base MTS cluster centroid.

(3) We combine *OmniTransfer* with six state-of-the-art (SOTA) MTS anomaly detection models and conduct experiments with real-world dataset from a top-tier enterprise. The results show that *OmniTransfer* reduces the initialization time by 59.72% and the training cost by 85.01% on average while maintaining high accuracy in detecting anomalies. Furthermore, we make our source code and the labeled datasets publicly available [14] to make it easier for readers to understand our work.

## II. Background

### A. MTS anomaly detection and services changes

In large-scale Web services, each entity is monitored on multiple metrics and these metrics are collected continuously at equal-space timestamps. The collected data of each entity forms an MTS with M metrics and N time points as a matrix $X \in R^{M \times N}$. For each time $t$, it is necessary to determine whether $X_t \in R^M$ is an anomaly. To comprehensively leverage the contextual information of MTS, we use its historical data $X_h = (X_{t-W}, X_{t-W+1}, ..., X_{t-1})$ of length $W$ to assist in identifying whether $X_t$ is an anomaly.

Due to the rapid expansion of the Internet, the release, upgrade, and configuration modification of Web services are becoming more frequent [15]–[18], resulting in changes to the services' running status and MTS patterns. These changes can lead to unexpected changes in MTS patterns, such as significant drops in page visits and machine performance metrics caused by page access failures and software errors. While other changes, such as less traffic and lower CPU usage due to configuration modifications, are expected. It is important to use an updated model for the changed entity because using an outdated model can result in many false alarms. To keep up with these frequent changes, it is necessary to reduce the model initialization time and ensure the accuracy of anomaly detection.

### B. Transfer learning

Transfer learning is a promising machine learning methodology that transfers knowledge across domains [19]. It can assist in training a target model with limited data by utilizing the knowledge from a source domain with sufficient data. Many works adopt the parameter-transfer approach [20].

However, fully transferring parameters may lead to negative transfer due to the differences in the prior distributions of the source and target domains [21]. To address this, AT-GP [21] and AnoTransfer [13] propose an adaptive transfer strategy to avoid negative transfer during the transfer learning process and achieve better generalizability.

### C. Related Work

*1) MTS clustering:* **MTS clustering based on traditional methods.** SPCA+AED [22] proposes a hybrid method based on the principal component analysis (PCA) similarity factor (SPCA) and the average-based Euclidean distance (AED). After the SPCA stage, lots of important information is lost.

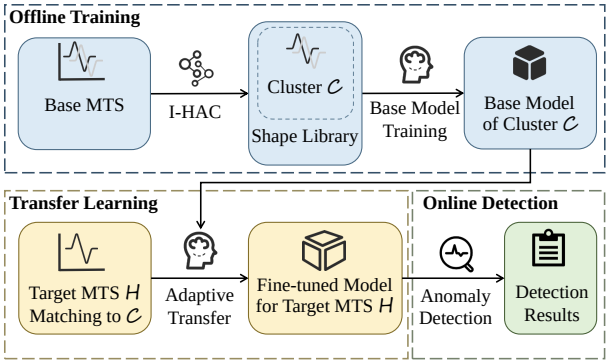


Fig. 3: The overview of *OmniTransfer*.

Toeplitz Inverse Covariance-Based Clustering (TICC) [23] mainly focuses on the MTS subsequences clustering. It is challenging for TICC to deal MTS with more than 100 time points (about one day). Besides, these methods usually can not handle the interference of anomalies, noises and phase shifts.

**MTS clustering based on deep learning.** CTF [24] clusters the low-dimensional features extracted by the pre-trained deep-learning model. OmniCluster [12] compresses MTS with a one-dimensional convolutional auto-encoder (AE). However, these low-dimensional features lose much information and are usually relevant to subsequent tasks. Moreover, training deep learning-based models requires significant computing resources. To overcome these limitations, we propose a task-agnostic clustering method, which ensures the efficiency, effectiveness, and robustness of clustering.

*2) MTS anomaly detection:* **MTS anomaly detection models based on deep learning.** USAD takes advantage of the ability to isolate anomalies of generative adversarial networks (GAN) and the stability of AE. DAGMM uses an AE to generate the low-dimensional features and reconstruction errors and feeds them into the Gaussian mixture model (GMM) to get the anomaly score. These two models are the first type, which mainly consists of some fully connected layers. TranAD uses a sequence encoder with self-attention to shorten the inference time. OmniAnomaly uses the recurrent neural network (RNN) and variational auto-encoder (VAE) structure to model the temporal dependence and stochasticity in MTS. InterFusion adopts the structure of RNN, convolutional neural network (CNN) and VAE, and employs a two-view embedding to explicitly learn the inter-metric and temporal dependencies. GDN is a prediction-based model which uses a graph neural network (GNN) to model the correlation between metrics. The above four models are the second type, which consists of specialized layers such as RNN, CNN, GNN, and attention. The specialized layers can capture more effective features for anomaly detection. CNN, GNN and attention help capture inter-metric dependence, while RNN can capture the temporal dependence of MTS. However, the above models face high training overhead when dealing with large-scale MTS data and long initialization time.

**MTS anomaly detection framework to reduce training overhead and model initialization time.** CTF provides a framework to reduce training overhead for RNN+VAE models [5], but it is not universal to other models. OmniCluster is a model-agnostic framework that can reduce the training overhead. It trains a model for each cluster and directly uses it for anomaly detection. However, it performs poorly when the shape of the target MTS and the cluster centroid differs. JumpStarter [11] uses compressed sensing to reduce the model initialization time. However, due to only using short-term data without sufficient inference and a simple model structure, it can not capture complex patterns and long temporal dependence.


Fig. 4: The overall process of I-HAC.

## III. Approach

### A. Overview

We propose a model-agnostic framework, named *OmniTransfer*, for MTS anomaly detection to reduce initialization time and training overhead. Fig.3 shows the overview of *OmniTransfer*, which includes three main stages: offline training, transfer learning, and online detection.

The offline training stage comprises two steps: improved hierarchical agglomerative clustering (I-HAC) and base model training. Fig. 4 illustrates the process of I-HAC, which can reduce interference from anomalies, noises and phase shifts. Thus, I-HAC can group MTS with similar shapes, addressing the first challenge. In the base model training stage, *OmniTransfer* trains a base model that can be used for transfer learning by using several MTS segments near the cluster centroid.

The target MTS undergoes transfer learning and online detection stages sequentially. First, we match the short-term data of the target MTS to an appropriate cluster and then use an adaptive transfer strategy to fine-tune the corresponding base model. The adaptive transfer strategy selects the best transfer strategy based on the distance between the target MTS and its corresponding cluster centroid. Finally, in the online detection stage, we use the fine-tuned model to detect anomalies in the target MTS.

### B. Offline training

*1) Improved hierarchical agglomerative clustering:* The I-HAC (illustrated in Fig. 4) aims to reduce the diversity of MTS and thus lower the training overhead of anomaly detection models. The specific steps of I-HAC are as follows:

*Baseline extraction.* By extracting the baseline, we can reduce the diversity of MTS patterns caused by noise and anomalies, as mentioned in the first challenge. Noise and anomalies can significantly impact the normal pattern of MTS. We extract the baselines (normal patterns) of MTS by removing extreme values and applying a moving average. Extreme values are more likely to be anomalies and their ratio is often

less than 5% [12], [13], [25]. Therefore, I-HAC removes the top 5% data that deviates from the mean value and then uses linear interpolation to fill the vacancies. Then, I-HAC applies the moving average to reduce the impact of noise.

*Phase alignment.* After extracting the baseline, we slice MTS into short-term segments, denoted as $\mathbf{MTS}_{seg} \in R^{M \times n}$, that match the length of the target MTS. We then align the phase shift.

First, we get the pivot $\mathbf{PVT}$ of the entire offline segments $\mathcal{D}$ according to $\mathbf{PVT} = \arg\min_{\mathbf{A} \in \mathcal{D}} \sum_{\mathbf{B} \in \mathcal{D}} Euc(\mathbf{A}, \mathbf{B})$. The Euclidean distance between two $\mathbf{MTS}_{seg}$ can be calculated by: $Euc(\mathbf{A}, \mathbf{B}) = (\mathbf{A} - \mathbf{B})^2$. Next, we use normalized cross-correlation ($NCC$) to estimate the best phase shift for all $\mathbf{MTS}_{seg}$ to align to $\mathbf{PVT}$. $s \in [-n+1, n-1]$ denotes the possible phase shifts. To retain short-term information, we use (1) to wrap round MTS.

$$\begin{aligned} &\mathbf{A}(s) = (\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n) \\ &\mathbf{B}(s) = \begin{cases} (\mathbf{B}_{n-s+1}, \ldots, \mathbf{B}_n, \mathbf{B}_1, \ldots, \mathbf{B}_{n-s}) & s \geq 0, \\ (\mathbf{B}_{-s+1}, \ldots, \mathbf{B}_n, \mathbf{B}_1, \ldots, \mathbf{B}_{-s}) & s < 0. \end{cases} \end{aligned} \quad (1)$$

$NCC$ reaches the maximum value when $s$ is close to the real phase shift, which is given by: $NCC(\mathbf{A}, \mathbf{B}, s) = \sum_{j=1}^{M} \frac{\sum_{i=1}^{n} \mathbf{A}(s)_i^j \cdot \mathbf{B}(s)_i^j}{||\mathbf{A}(s)^j||_2 \cdot ||\mathbf{B}(s)^j||_2}$. The best phase shift $s^*$ obtained by: $s^* = \arg \max_{s \in [-n+1, n-1]} NCC(\mathbf{PVT}, \mathbf{MTS}_{seg}, s)$. Finally, we align the phase shift $s^*$ of $\mathbf{MTS}_{seg}$ to get $\mathbf{MTS}_{seg}^{'}$.

*Clustering. OmniTransfer* gets the clustering result using hierarchical agglomerative clustering (HAC) and the Euclidean distance. HAC with average linkage is adopted for the following reasons. (1) The HAC algorithm is robust to the extreme value because it clusters on the rank of distances rather than the value. (2) Each data in the cluster have the same effect on the distance measure, making the distance measure transitive. After clustering, several segments near the cluster centroid are saved for base model training and matching the target MTS.

*2) Base model training:* The VAE-based algorithms [5], [7] model the relationship between the latent variable $z$ and the observed variable $x$. They typically train their models by optimizing the Evidence Lower Bound (ELBO): $\mathcal{L}_1 = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}[q_\phi(z|x)||p_\theta(z)]$, which is comprised of a reconstruction probability and a regularization term. $p_\theta$ is a generative model that represents the real posterior of the data, while $q_\phi$ is an inference model aiming to estimate the posterior. The $D_{KL}$ term represents the Kullback-Leibler divergence [26]. On the other hand, AE-based and prediction-based models [6], [8]–[10] focus on reconstructing or predicting the target. These models trained by minimizing the mean square error (MSE) between the target and output: $\mathcal{L}_2 = \mathbb{MSE}(target - output)$.

### C. Transfer learning

*1) Transfer preparations:* To train the target model for each target MTS, *OmniTransfer* utilizes a base model $E$, which is selected based on its cluster centroid's proximity to the target short-term data $\mathbf{H} \in R^{M \times n}$. First, we perform baseline extraction and phase alignment to get $\mathbf{H}^{'}$. Then, we calculate the distance between $\mathbf{H}^{'}$ and the centroid of each cluster and select the closest one and its corresponding base model for transfer learning. We use $\mathbf{H}$ to fine-tune the base model.

*2) Adaptive transfer strategy:* We propose an adaptive transfer strategy that automatically selects whether to transfer full parameters or partial parameters for each target MTS. When the target MTS and the nearest cluster centroid are relatively similar, we use the full parameter transfer strategy and fine-tune the entire base model's parameters directly. Otherwise, we employ the partial parameter transfer strategy. Specifically, we initialize a target model with random parameters and load part of the base model's parameters into the target model. First, we update the remaining parameters while keeping the transferred parameters fixed. Then we fine-tune all of the parameters of the target model.

*3) Transfer layer selection:* We adopt the partial parameter transfer strategy when there is a significant difference between the target MTS and its corresponding cluster centroid. We select specific layers based on their capabilities and characteristics for transferring. As mentioned in § II-C2, these SOTA MTS anomaly detection models fall into two categories based on their structures. For the former type, their outer layers focus on more extensive tasks and capture more generic features [27]–[29], while the inner layers are designed to capture more task-specific features [30], [31]. For the latter, the specialized layers (e.g., RNN, CNN, attention, and GNN) capture more generic features, while the fully connected layers focus more on specific tasks [24], [32]–[35]. It is recommended to transfer the parameters of the outer layers or the specialized layers when adopting the partial parameter transfer strategy, as they learn generic features that are often not specific to a particular task.

### D. Online detection

We use the fine-tuned model for online detection. For the VAE-based models, their anomaly score corresponds to the negative reconstruction probability, which is given by: $\mathcal{AS}_1 = -\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$. $\log p_\theta(x|z)$ denotes the reconstruction probability of each observed variable $x$. The smaller the reconstruction probability, the greater the probability that the data point is an anomaly. For the AE-based models and prediction-based models, we calculate the anomaly scores according to $\mathcal{AS}_2 = \mathbb{MSE}(target - output)$, which measures the difference between the target and the output. A greater difference indicates a higher probability that the data point is an anomaly.

## IV. Evaluation

In this section, we introduce the experimental setup, including dataset, evaluation metrics, hyperparameters and experiment environment of *OmniTransfer*. Then, we conduct extensive experiments to evaluate the performance of *OmniTransfer* and answer the following research questions:

RQ1. How does the effectiveness and efficiency of *OmniTransfer* compare to baseline methods?

RQ2. How much initialization time can *OmniTransfer* reduce compared to non-transfer learning methods?

RQ3. How much do the key techniques contribute to its overall performance?

### A. Experimental setup

*1) Dataset and environment:* In this work, we use the server data from one of the world's largest short video companies. It contains 400 entities, and each entity has 19 metrics and persists for seven days. We use the last two days as test data, and the fifth day as training data for transfer learning. In practice, the proportion of pattern changes such as adding or upgrading is relatively small. To better evaluate the algorithm, we randomly choose 50% of the entities for offline training, and the remaining 50% simulate change data for online detection. The online data is labeled by experienced operators. Please note that we only choose 400 entities from millions for evaluation since the labeling work is time-consuming. By the way, we do not use public datasets (e.g., SWaT, WADI [36], SMD [5], SMAP and MSL [37]), mainly because the number of entities is too small (less than 55 entities). All experiments are run on a server with two 16C32T Intel(R) Xeon(R) Gold 5218 CPU @ 2.30 GHz, one NVIDIA(R) Tesla(R) V100S, and 192 GB RAM.

*2) Evaluation metrics: OmniTransfer* outputs an anomaly score for each point and determines whether it is an anomaly by a threshold. Thus MTS anomaly detection can be regarded as a binary classification problem. We use the $F_1$ to evaluate the effectiveness, which is given by $Precision = \frac{TP}{TP+FP}, Recall = \frac{TP}{TP+FN}, F_1 = 2 \times \frac{Precision \times Recall}{Precision+Recall}$. $TP$ represents True Positives, $FP$ represents False Positives, and $FN$ represents False Negatives. The $F_1$ is obtained using the micro-average method. By enumerating all possible thresholds, we obtain the best $F_1$ for each model, denoted by $F_1^*$. Additionally, we record the time required for model training to evaluate efficiency.

*3) Hyperparameters:* We use the best empirical values for most parameters based on experimental results. Specifically, We set the sliding window length for the moving average to 12. We use five segments closest to the centroid for each cluster to train the base models and use a sliding window with a length of 60.

### B. OmniTransfer vs. baseline models (RQ1)

We combine *OmniTransfer* with six typical unsupervised MTS anomaly detection methods: OmniAnomaly, InterFusion, DAGMM, USAD, GDN, and TranAD. These models focus on different challenges in MTS anomaly detection and have different structures. To demonstrate the effectiveness and efficiency of *OmniTransfer*, we compare it with OmniCluster [12], one model/entity, CTF [24], and JumpStarter [11]. *OmniTransfer*, OmniCluster, and one model/entity are model-agnostic training frameworks or strategies that can be combined with various models. The results of these methods are presented at the top of Table I. However, CTF is designed specifically for the RNN+VAE model. And JumpStarter, which doesn't require training, cannot be combined with *OmniTransfer*. The results of these two baselines are shown at the bottom of Table I. *OmniTransfer* outperforms all baselines in effectiveness and is more efficient than all baseline models except for OmniCluster. We will try to analyze the reasons for this result in detail.

TABLE I: Overall performance.

| Model | *OmniTransfer* | | OmniCluster | | one model/entity | |
|---|---|---|---|---|---|---|
| | $F_1^*$ | Time (s) | $F_1^*$ | Time (s) | $F_1^*$ | Time (s) |
| OmniAnomaly | **0.8865** | 1212.99 | 0.5169 | 560.47 | 0.7000 | 9888.25 |
| InterFusion | **0.8666** | 1585.63 | 0.5830 | 566.56 | 0.4769 | 8884.94 |
| DAGMM | **0.8375** | 244.48 | 0.7104 | 137.37 | 0.8245 | 2947.47 |
| USAD | **0.8222** | 80.16 | 0.7468 | 109.04 | 0.7875 | 691.77 |
| GDN | **0.8026** | 54.55 | 0.6806 | 42.81 | 0.7405 | 265.27 |
| TranAD | **0.8995** | 114.53 | 0.7797 | 102.10 | 0.8538 | 591.67 |
| JumpStarter | 0.4211 | 4786.67 | - | - | - | - |
| CTF | 0.8661 | 4965.61 | - | - | - | - |

*1) Compare with OmniCluster: OmniTransfer* outperforms OmniCluster by 10.10% to 71.50%. *OmniTransfer* retains the information of short-term data and considers the problem of phase shifts. In contrast, OmniCluster compresses MTS in the temporal dimension and removes some metrics, resulting in a loss of shape and metric information. *OmniTransfer* uses transfer learning to train a suitable model for each MTS, whereas OmniCluster trains a base model for each cluster. The training time of *OmniTransfer* is 64.56% higher than OmniCluster. Because OmniCluster only trains base models without fine-tuning.

*2) Compare with one model/entity:* In terms of $F_1$, *OmniTransfer* achieves an average improvement of 21.35%, and it reduces the training overhead by 85.01%. One model/entity uses only short-term MTS to train model, which is insufficient for deep learning-based models. Moreover, training the model from scratch usually takes longer to converge. Therefore, the performance and efficiency of one model/entity strategy are unsatisfactory. In contrast, *OmniTransfer* performs better by maximizing the use of the base MTS to train the base model. The overall training overhead of *OmniTransfer* benefits from only a small number of base models that need to be trained and the base models help accelerate the convergence of the target model training.

*3) Compare with CTF:* CTF is specifically designed for RNN+VAE models, particularly for OmniAnomaly. Therefore we only compare the performance of *OmniTransfer*+OmniAnomaly with CTF. The $F_1$ of *OmniTransfer*+OmniAnomaly is approximately 2.4% higher than CTF. CTF produces fine-tuned models at the cluster level, which cannot be deployed perfectly to each MTS. The training time of CTF is more than four times that of *OmniTransfer*+OmniAnomaly. This is because CTF fine-tunes cluster-level models based on a dataset-level pre-trained model. As the difference between the source domain and the target domain of CTF is significant, it requires more MTS and training epochs during fine-tuning.

*4) Compare with JumpStarter:* JumpStarter successfully reduces model initialization time, but its $F_1$ is significantly lower and the training time is much longer compared to *Om-*

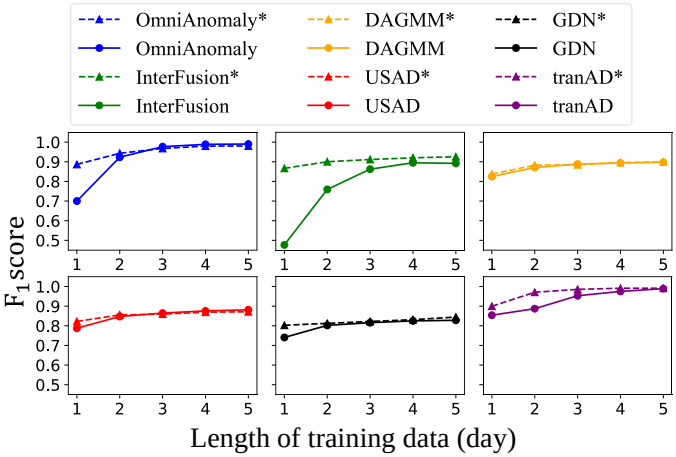


Fig. 5: The performance of *OmniTransfer* and one model/entity with different initialization time. '*' denotes the corresponding result of combining *OmniTransfer* and without * denotes the result of one model/entity strategy.

*niTransfer*. JumpStarter uses only short-term data to sample and reconstruct the normal value, which is usually sufficient. And the outlier-resistant sampling method may not always successfully remove anomaly points in highly volatile metrics, limiting the performance of JumpStarter. Additionally, the complicated sampling process in JumpStarter increases the training time seriously.

### C. Effect on reducing model initialization time (RQ2)

In this section, we conduct experiments on six anomaly detection models to verify the effect of *OmniTransfer* in reducing model initialization time. We increase the initialization time from one day to five days. Fig. 5 demonstrates that *OmniTransfer* outperforms one model/entity by 7.99% with one day and two days of training data on average. And *OmniTransfer* using two days of training data performs almost the same as one model/entity using all training data, highlighting its ability to reduce the model initialization time.

### D. Ablation experiment (RQ3)

To demonstrate the effect of three key technologies in *OmniTransfer*: (1) clustering; (2) phase alignment; (3) transfer learning. We reconfigure *OmniTransfer* to create three variants. C1: Only one base model is used for transfer learning, and the data used to train the base model are randomly selected. C2: Do not align the phase shift. C3: The base model is directly used for anomaly detection of all MTS in the cluster. Table II shows the results of each variant.

**Effect of clustering.** With an $F_1$ of lower than 0.65, the performance of C1 is far from satisfactory. The large difference between the base MTS and the target MTS makes transfer learning challenging. While clustering can effectively group MTS with similar shapes for base model training, making it easy to transfer the knowledge of base MTS to target MTS.

**Effect of phase alignment.** C2 needs more training overhead and has a poor performance than *OmniTransfer*. Without phase alignment, the diversity of MTS patterns increases, resulting in more clusters, and the training overhead increases dramatically. Additionally, it is difficult to match the target data with the appropriate cluster without phase alignment.

**Effect of transfer learning.** C3 directly uses the base model of each cluster for anomaly detection. Although the target MTS should be reasonably similar to its matching cluster centroid, there are still many tiny differences. These differences make the $F_1$ relatively poor.

TABLE II: Ablation experiment.

| Model | *OmniTransfer* | C1 | C2 | C3 |
|---|---|---|---|---|
| OmniAnomaly | **0.8865** | 0.6925 | 0.7979 | 0.7242 |
| InterFusion | **0.8666** | 0.656 | 0.7668 | 0.7319 |
| DAGMM | **0.8375** | 0.7966 | 0.8071 | 0.7804 |
| USAD | **0.8222** | 0.7754 | 0.7928 | 0.8008 |
| GDN | **0.8026** | 0.7702 | 0.7647 | 0.7643 |
| TranAD | **0.8995** | 0.8805 | 0.8884 | 0.8436 |

## V. Discussion

We have some ideas for future work. (1) The same ideas and key techniques of *OmniTransfer* can reduce model initialization time and training overhead for other tasks, such as the prediction and classification of large-scale MTS. (2) The threshold for adaptive transfer strategy selection is currently selected based on experience. The threshold selection method can be further studied. (3) The clustering method is not the focus of this paper, and the clustering method for MTS is also a popular research direction.

## VI. Conclusion

This paper clearly points out the limitations of existing methods in large-scale MTS scenarios. And we propose *OmniTransfer*, a model-agnostic, unsupervised, and efficient anomaly detection framework for several SOTA MTS anomaly detection models. It uses transfer learning to reduce model initialization time and training overhead effectively. We propose I-HAC to improve the effect and efficiency of transfer learning. Our experiments use a real-world dataset from a large Web content service provider. The results show that *OmniTransfer* can reduce the initialization time by 59.72% and improve training efficiency by 85.01% compared to baseline models. We believe *OmniTransfer* is useful for large Web services, especially when monitoring millions of services that change frequently. *OmniTransfer* makes the anomaly detection models as fast and cost-effective as possible for the large-scale and changing MTS.

## Acknowledgment

We thank Yue Li, Xiaoteng Pan, Shiqi Chen, Jiaran Zhang and Yizhen Zhang for their contributions to this work. This work was supported in part by the Advanced Research Project of China (No. 31511010501), National Natural Science Foundation of China (Grant No. 62272249, 62072264), and Natural Science Foundation of Tianjin (Grant No. 21JCQNJC00180).

## REFERENCES


[1] M. Ma, J. Xu, Y. Wang, P. Chen, Z. Zhang, and P. Wang, "Automap: Diagnose your microservice-based web applications automatically," in *Proceedings of The Web Conference 2020*, ser. WWW '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 246–258.

[2] Y. Su, Y. Zhao, M. Sun, S. Zhang, X. Wen, Y. Zhang *et al.*, "Detecting outlier machine instances through gaussian mixture variational autoencoder with one dimensional cnn," *IEEE Transactions on Computers*, vol. 71, no. 4, pp. 892–905, 2022.

[3] F. Ye, Z. Lin, C. Chen, Z. Zheng, and H. Huang, "Outlier-resilient web service qos prediction," in *Proceedings of the Web Conference 2021*, ser. WWW '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 3099–3110.

[4] X. Zhang, J. Kim, Q. Lin, K. Lim, S. O. Kanaujia, Y. Xu *et al.*, "Cross-dataset time series anomaly detection for cloud systems," in *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, 2019, pp. 1063–1076.

[5] 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 *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2828–2837.

[6] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "Usad: Unsupervised anomaly detection on multivariate time series," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 3395–3404.

[7] Z. Li, Y. Zhao, J. Han, Y. Su, R. Jiao, X. Wen *et al.*, "Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 3220–3230.

[8] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho *et al.*, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *International Conference on Learning Representations*, 2018.

[9] S. Tuli, G. Casale, and N. R. Jennings, "Tranad: Deep transformer networks for anomaly detection in multivariate time series data," *arXiv preprint arXiv:2201.07284*, 2022.

[10] A. Deng and B. Hooi, "Graph neural network-based anomaly detection in multivariate time series," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 5, 2021, pp. 4027–4035.

[11] M. Ma, S. Zhang, J. Chen, J. Xu, H. Li, Y. Lin *et al.*, "{Jump-Starting} multivariate time series anomaly detection for online service systems," in *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, 2021, pp. 413–426.

[12] S. Zhang, D. Li, Z. Zhong, J. Zhu, M. Liang, J. Luo *et al.*, "Robust system instance clustering for large-scale web services," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 1785–1796.

[13] S. Zhang, Z. Zhong, D. Li, Q. Fan, Y. Sun, M. Zhu *et al.*, "Efficient kpi anomaly detection through transfer learning for large-scale web services," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 8, pp. 2440–2455, 2022.

[14] 2023. [Online]. Available: https://anonymous.4open.science/r/OmniTransfer

[15] M. Ma, S. Zhang, D. Pei, X. Huang, and H. Dai, "Robust and rapid adaption for concept drift in software system anomaly detection," in *2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)*, 2018, pp. 13–24.

[16] Z. Li, Q. Cheng, K. Hsieh, Y. Dang, P. Huang, P. Singh *et al.*, "Gandalf: An intelligent, End-To-End analytics service for safe deployment in Large-Scale cloud infrastructure," in *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*. Santa Clara, CA: USENIX Association, Feb. 2020, pp. 389–402.

[17] Y. Zhang, J. Yang, Z. Jin, U. Sethi, K. Rodrigues, S. Lu *et al.*, "Understanding and detecting software upgrade failures in distributed systems," in *Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles*, ser. SOSP '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 116–131.

[18] N. Zhao, J. Chen, Z. Yu, H. Wang, J. Li, B. Qiu *et al.*, "Identifying bad software changes via multimodal anomaly detection for online service systems," in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ser. ESEC/FSE 2021. New York, NY, USA: Association for Computing Machinery, 2021, p. 527–539.

[19] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu *et al.*, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2021.

[20] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[21] B. Cao, S. J. Pan, Y. Zhang, D.-Y. Yeung, and Q. Yang, "Adaptive transfer learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 24, no. 1, pp. 407–412, Jul. 2010.

[22] J. Wu, S. Nguang, J. Shen, G. Liu, and Y. Li, "Robust h$\infty$ tracking control of boiler–turbine systems," *ISA transactions*, vol. 49, no. 3, pp. 369–375, 2010.

[23] D. Hallac, S. Vare, S. Boyd, and J. Leskovec, "Toeplitz inverse covariance-based clustering of multivariate time series data," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 215–223.

[24] M. Sun, Y. Su, S. Zhang, Y. Cao, Y. Liu, D. Pei *et al.*, "Ctf: Anomaly detection in high-dimensional time series with coarse-to-fine model transfer," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[25] 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)*, 2018, pp. 1–10.

[26] J. M. Joyce, "Kullback-leibler divergence," in *International encyclopedia of statistical science*. Springer, 2011, pp. 720–722.

[27] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014.

[28] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson, "Factors of transferability for a generic convnet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 9, pp. 1790–1802, 2016.

[29] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway *et al.*, "Convolutional neural networks for medical image analysis: Full training or fine tuning?" *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[30] G. Vrbančič and V. Podgorelec, "Transfer learning with adaptive fine-tuning," *IEEE Access*, vol. 8, pp. 196 197–196 211, 2020.

[31] Y. Guo, H. Shi, A. Kumar, K. Grauman, T. Rosing, and R. Feris, "Spottune: Transfer learning through adaptive fine-tuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[32] R. K. Samala, H.-P. Chan, L. Hadjiiski, M. A. Helvie, C. D. Richter, and K. H. Cha, "Breast cancer diagnosis in digital breast tomosynthesis: Effects of training sample size on multi-stage transfer learning using deep neural nets," *IEEE Transactions on Medical Imaging*, vol. 38, no. 3, pp. 686–696, 2019.

[33] E. Cetinic, T. Lipic, and S. Grgic, "Fine-tuning convolutional neural networks for fine art classification," *Expert Systems with Applications*, vol. 114, pp. 107–118, 2018.

[34] M. A. Humayun, H. Yassin, J. Shuja, A. Alourani, and P. E. Abas, "A transformer fine-tuning strategy for text dialect identification," *Neural Computing and Applications*, pp. 1–10, 2022.

[35] K. Lu, A. Grover, P. Abbeel, and I. Mordatch, "Frozen pretrained transformers as universal computation engines," 2022.

[36] A. P. Mathur and N. O. Tippenhauer, "Swat: a water treatment testbed for research and training on ics security," in *2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*, 2016, pp. 31–36.

[37] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding," *arXiv e-prints*, p. arXiv:1802.04431, Feb. 2018.