Unsupervised anomaly detection under a multiple modeling strategy via model set optimization through transfer learning

Masanao Natsumeda*, Takehiko Mizoguchi*,
*NEC Corporation
Tokyo, Japan.
Email: {mnatsumeda, tmizoguchi}@nec.com

Wei Cheng†, Yuncong Chen† and Haifeng Chen†
†NEC Laboratories America, Inc.
NJ, USA.
Email: {weicheng, yuncong, haifeng}@nec-labs.com

***Abstract*—Unsupervised anomaly detection approaches have been widely accepted in applications for industrial systems. Industrial systems often operate with multiple modes since they work for multiple purposes or under different conditions. In order to deal with the difficulty of anomaly detection due to multiple operating modes, multiple modeling strategies are employed. However, estimating the optimal set of models is a challenging problem due to the lack of supervision and computational burden. In this paper, we propose DeconAnomaly, a deep learning framework to estimate the optimal set of models using transfer learning for unsupervised anomaly detection under a multiple modeling strategy. It reduces computational burden with transfer learning and optimizes the number of models based on a surrogate metric of detection performance. The experimental results show clear advantages of DeconAnomaly.**

***Index Terms*—multivariate timeseries, anomaly detection, deep learning**

I. Introduction

Unsupervised anomaly detection approaches have been widely accepted in applications for industrial systems since they do not require historical data from failures [1]. While impacts from failures in industrial systems can be catastrophic, they rarely happen due to the safety functions and dedication of engineers and operators. This results in a lack of labeled anomalies and necessitates using unsupervised anomaly detection approaches in many cases.

Industrial systems often operate with multiple modes since they work for multiple purposes or under different conditions. For example, in chemical plants, they are attributed to changes in production strategies, variations in product specifications, or fluctuations in the external environment [2], [3]. In addition, start-up and shut-down operations also have their operating modes. Industrial systems can behave differently among different operating modes. They appear as different behaviors in sensor readings, and their difference can be significant. It results in many false alarms or missing alarms when global modeling approaches, which construct a model for all the modes, are employed.

In order to deal with the difficulty of anomaly detection due to multiple operating modes, multiple-modeling strategies are employed [4]–[7]. The concept of the multiple-modeling strategies is straightforward and intuitive. Under the strategies, a model for each operating condition is constructed, and the system is monitored while switching between models based on the operating mode.

Although the multiple modeling strategies work for major operating modes, they bring two issues. One is an increase in false alarms from minor operating modes due to limited data for training. The other is an increase in complexity at operation. The models are switched to match the current operating mode. In addition, it is labor-intensive to maintain a large number of models.

The situation becomes more complicated for specific operating modes. For example, start-up or shut-down operations of industrial plants are viewed as a chain of events. Each event can form an operating mode. In addition, not all of their sub-systems are active during the operations. Dependencies between sub-systems change over time during the operations, and some sub-systems are independent until certain operations become independent in the middle of the operations. A simple solution is training models for all combinations of a sub-system and an operating mode. However, this results in a large number of models for monitoring.

In order to reduce the number of models, a model should cover multiple operating modes, multiple sensor groups associated with sub-systems, or both; however, the assignment is not trivial since the optimal set of models is needed to be estimated without labeled anomalies. In addition, not only we cannot measure the performance of each combination of operating modes and sensor groups until the corresponding model is trained, but also there are a lot of possible combinations.

The contributions of this paper are summarized as follows:

- We propose DeconAnomaly, a deep learning framework to estimate the optimal set of models using transfer learning for unsupervised anomaly detection under a multiple modeling strategy. To the best of our knowledge, DeconAnomaly is the first deep learning-based framework for multivariate time series anomaly detection which explicitly explores the optimal combination of sensor groups and operating modes.
- We evaluate DeconAnomaly with five datasets and verify

that exploring the optimal set of models improves detection performance while reducing the number of models. With a penalty for a larger number of models, the number of models is reduced by about 70% at least.
- We release the VAM[1] dataset to the public for better reproducibility and subsequent studies for anomaly detection.

II. Preliminaries

In this section, we first provide the problem definition and then review related works.

A. Problem definition

Our main goal is to obtain the optimal set of models monitoring its target system without labeled anomalies. The target system has S sensor groups and O operating modes. The number of combinations of a sensor group and an operating mode is $S \times O$.

We define the region r_{ij} as a pair of the ith sensor group and the jth operating mode. A model covers a set of regions which is called a territory.

Let M be the number of candidate models, $s_d^{(m)}$ be a detection score of the mth candidate model, λ be the penalty score to have a model, and $z^{(m)} \in \{0, 1\}$ be an indicator function of selecting the mth model as a member of the optimal set of models.

The problem we aim to study is to get a set of models which solves the optimization problem defined as:

$$\text{minimize} - \sum_{m=1}^{M} s_d^{(m)} z^{(m)} + \lambda \sum_{m=1}^{M} z^{(m)}, \tag{1}$$

$$s.t. \sum_{m=1}^{M} w_{ij}^{(m)} z^{(m)} = 1, \tag{2}$$

where $w_{ij}^{(m)} \in \{0, 1\}$ is an indicator function of occupancy at the region r_{ij} by the mth model.

B. Deep learning for unsupervised anomaly detection in multivariate time series

Due to the importance of temporal features for anomaly detection [8], deep learning-based methods for anomaly detection employ sequence modeling. They are grouped into forecasting-based methods [9], [10], and reconstruction-based methods [11], [12]. Forecasting-based methods detect anomalies based on prediction errors. LSTM-AD [9] and LSTM-NDT [10] use stacked Long Short-Term Memory (LSTM) cells to get hidden representations of time series and use a fully connected layer to predict one-step ahead values from the representations. Reconstruction-based methods detect anomalies based on reconstruction errors. EncDec-AD [11] uses an LSTM Autoencoder, which has an LSTM cell to get hidden representations of input time series and an LSTM cell to reconstruct the time series based on the hidden representations. LSTM-VAE [13] combines LSTM with VAE to jointly model representations of input time series and its error distribution, taking temporal dependencies into account. OmniAnomaly [12] explicitly models temporal dependencies between stochastic variables to learn robust representations of input time series data.

DeconAnomaly uses a variant of LSTM Autoencoder similar to EncDec-AD since reconstruction-based methods do not require time series to be predictable. The reason is that time series from industrial systems are often unpredictable due to external factors. Unlike the LSTM Autoencoder used in EncDec-AD, the model consists of an independent neural network for each sensor group called lNet and a neural network connecting lNets called gNet so that they can be decomposed and recombined. Since DeconAnomaly is a framework rather than a method, it is also applicable to forecasting-based methods.

C. Knowledge transfer from pre-trained models

Many studies have shown pre-trained models contain helpful information to train neural networks [14], [15] or solve another task [16]–[18]. When parameters in a pre-trained model are used as initial values in a neural network, they improve the model's performance and speed up the convergence of learning [14], [15]. In particular, pre-training helps optimize the parameters of the lower-level layers [14], [16], which are closer to the input layer.

Task transferability is essential in transfer learning since incorporating unrelated tasks harms the performance. In [19], task transferability is measured based on validation errors between transferred models, and the resulting task transferability reduces the labeled training samples by two third while the performance is kept nearly the same. In [20] and [21], task transferability is efficiently measured based on the similarity of feature representations. They enable to estimate of task transferability without further training when source models are given. The above approaches train a source model for each task and then adapt it to the target task. Different from them, DeconAnomaly trains the source model for all of the tasks first and then adapts it to the target task in order to make similar or complemental tasks visible at pre-training.

III. DeconAnomaly

In this section, we elaborate on our proposal, DeconAnomaly. We introduce the framework of DeconAnomaly followed by its model architecture, the definition of the detection score, the penalty score, and the anomaly score, and the approach to determining the threshold for anomaly detection.

A. Framework

DeconAnomaly has two operational phases, i.e., training and testing. We assume operating modes are given in training and testing.

The training phase consists of five stages. In Stage I, the source model is trained for all the operating modes and sensor groups. In Stage Ⅱ, territories are sampled, and the source model is adapted to each territory within assigned

[1] github.com/mnatsumeda/deconanomaly

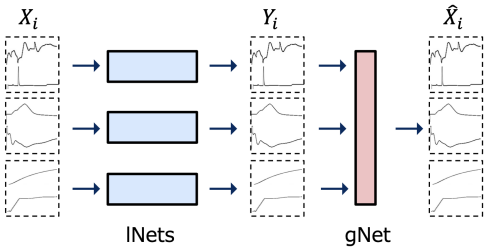

Fig. 1: Root model. Each lNet and the gNet are implemented as an LSTM-Autoencoder and a fully connected layer without intercept terms, respectively.

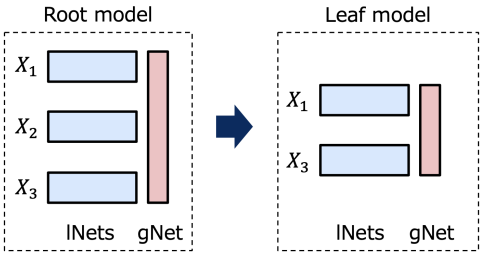

Fig. 2: Constructing a leaf model through model decomposition.

computational budgets. The sampling strategy is explained in Sec. IV-B since it is case specific. During the adaptation, the detection performance of each adapted model is also estimated. In Stage Ⅲ, the optimal set of models is estimated based on the detection performance. In Stage Ⅳ, models in the optimal set are fine-tuned. In Stage V, the threshold of anomaly scores is computed for each model in the optimal set.

During the testing phase, models are chosen based on the operating mode. Each model for the current operating mode computes an anomaly score after a new observation is given. If any of anomaly scores exceeds its corresponding threshold, an anomaly is detected, and the corresponding sensor groups are regarded as anomalous.

B. Root model

The model architecture of DeconAnomaly consists of two types of neural networks, namely lNets and a gNet. The source model is called the root model since it covers all the sensor groups with lNets and a gNet as shown in Fig. 1. Each lNet is dedicated to its assigned sensor group, and all the lNets are weakly coupled through the gNet. This architecture enables to decompose of the model for a sub-model of some sensor groups A lNet is implemented as an LSTM-Autoencoder, and a gNet is implemented as a fully connected layer without intercept terms so that not only parameters in lNets but also those in the gNet are separable to each sensor group.

The root model takes a multivariate time series as its input, and each lNet takes part of the time series corresponding to its sensor group. Each lNet reconstructs its input time series by itself. Outputs from lNets are concatenated and fed into the gNet to let the gNet improve the reconstruction taking the dependencies among sensor groups into account.

Let $\boldsymbol{X}_i$ be a multivariate time series for ith sensor group, $\mathcal{F}_i$ be its lNet, $\boldsymbol{Y}_i$ be the reconstruction of $\boldsymbol{X}_i$ by lNet $\mathcal{F}_i$, $\boldsymbol{Y} = [\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots, \boldsymbol{Y}_S]$ be the concatenated matrices of $\boldsymbol{Y}_i$, $\hat{\boldsymbol{X}} = [\hat{\boldsymbol{X}}_1, \hat{\boldsymbol{X}}_2, \ldots, \hat{\boldsymbol{X}}_S]$ be the output through gNet $\mathcal{G}$.

The outputs from the Root model $\hat{\boldsymbol{X}}$ is defined as:

$$\hat{\boldsymbol{X}} = \mathcal{G}(\boldsymbol{Y}), \tag{3}$$

where:

$$\boldsymbol{Y}_i = \mathcal{F}_i(\boldsymbol{X}_i). \tag{4}$$

The root model is trained with Mini Batch Gradient Descent. Let K be the number of time series segments in a mini-batch, $\boldsymbol{X}^{(i)}$ be the ith input time series segment in the mini-batch, $\boldsymbol{Y}^{(i)}$ be its reconstructed values with all the lNets, $\hat{\boldsymbol{X}}^{(i)}$ be its reconstructed values with all the lNets and the gNet.

The loss function of the root model L_{root} is defined as:

$$L_{root} = \frac{1}{K}\sum_{i=1}^{K} L_i + \frac{1}{K}\sum_{i=1}^{K} G_i, \tag{5}$$

where:

$$L_i = \|\hat{\boldsymbol{X}}^{(i)} - \boldsymbol{X}^{(i)}\|_2^2, \tag{6}$$

$$G_i = \|\boldsymbol{Y}^{(i)} - \boldsymbol{X}^{(i)}\|_2^2. \tag{7}$$

L_{root} is also used for performance evaluation on validation data during training.

The second term is Mean-Square Error (MSE) only by lNets, and it is called transferability constraint. This constraint encourages lNets to maintain a reasonably good reconstruction performance within each sensor group. It results in lNets keeping the information for the reconstruction in their parameters as much as possible, and parameters in the gNet only keep information on relatedness between sensor groups. In this way, lNets and the gNet are weakly coupled with each other and achieve better transferability.

C. Transfer learning with model decomposition

DeconAnomaly employs domain-specific fine-tuning as its transfer learning method similar to [17]. The root model covers all the regions and is adapted to its sub-territory. The adapted model is called a leaf model. In order to enable faster convergence at the transfer learning and avoid the negative transfer as much as possible, parameters in the root models are fully utilized.

Leaf models have the same model architecture as the root model, as shown in Fig. 2. Each consists of lNets and a gNet. Since lNets are independent and separable concerning sensor groups, lNets in the root model are extracted and applied to a leaf model for selected sensor groups. Although the gNet in the root model is connected to all of the lNets, the parameters correspond to the selected sensor groups because the gNet is a fully connected layer. The parameters are a square matrix whose rows and columns are associated with corresponding sensors. Its submatrix for sensors in the selected sensor groups

is the gNet. The parameters taken from the root model are used as the initial values at the transfer learning.

The loss function of a leaf model L_{leaf} is defined as:

$$L_{leaf} = \frac{1}{K}\sum_{i=1}^{K} L_i \tag{8}$$

where:

$$L_i = \|\hat{\boldsymbol{X}}^{(i)} - \boldsymbol{X}^{(i)}\|_2^2, \tag{9}$$

where K is the number of time series segments in a mini-batch. L_{leaf} is also used for performance evaluation on validation data during adaptation.

D. Estimating the optimal set of models

The optimization problem (1) defined in Sec. II is an NP-hard problem; however, its approximated solution can be efficiently obtained with the branch and bound algorithm. The parameters in the optimization problem are the detection scores and the penalty score.

The detection score of a leaf model is defined with the distribution of anomaly scores for each region in its territory. The IQR rule is employed to represent the distribution since it does not make any assumption on error distribution [22] and gives a relatively stable estimation with a small number of samples. Let t_{ij} be the feature value of region r_{ij} computed with corresponding validation data. The feature value t_{ij} is defined as:

$$t_{ij} = Q3_{ij} + 1.5(Q3_{ij} - Q1_{ij}), \tag{10}$$

where $Q3_{ij}$ is the upper quartile and $Q1_{ij}$ is the lower quartile. Based on the feature value, the detection score of the mth leaf model $s_d^{(m)}$ is defined as:

$$s_d^{(m)} = -\sum_{\langle i,j\rangle} t_{ij}, \tag{11}$$

where $\langle i, j\rangle$ represents the territory of the leaf model. Given a possible set of models for monitoring, the summation of the detection scores takes into account the feature value from each region. This approach enables comparison with different sets of models even if the number of models is different.

The penalty score can be viewed as a hyper-parameter that balances the models' performance and complexity. We propose its automatic determination strategy assuming the value of the objective function in 1 by the optimal set of models is smaller than the linear interpolation between the single model and the empirical best set without the penalty. The single model covers all the sensor groups and operating modes. The empirical best set is obtained as the solution of the optimization problem (1) without the second term.

Let S_{db} be the summation of the detection scores by the empirical best set, N_b be the number of models in the empirical best set and S_{ds} be the summation of the detection scores by the single model. The penalty score λ is defined as:

$$\lambda = (1+\varepsilon)\frac{S_{db} - S_{ds}}{N_b - 1}, \tag{12}$$

where ε is a quite small value to ensure the optimal set of models has a better value by the objective function in the optimization problem (1) than the corresponding value of the linear interpolation.

E. Reconstruction-based anomaly detection with the root model or a leaf model

An anomaly score is computed based on reconstruction errors. It usually incorporates all the reconstruction errors for the input time series segment [11]. However, it is less sensitive to anomalies since it averages various errors in the time series segment. In order to keep the model sensitivity high, reconstruction errors at the latest time within the time series segment is used to compute an anomaly score of the segment. Let $\boldsymbol{x}^{(t)} \in \mathbb{R}^D$ be the latest observations at the t^{th} time series segment and $\hat{\boldsymbol{x}}^{(t)}$ be its reconstructed values. The anomaly score a_t is defined as:

$$a_t = \frac{1}{D}\|\hat{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|_2^2. \tag{13}$$

The threshold of anomaly score is determined with validation data based on the Peaks-Over-Threshold (POT) approach [12], [23]. The POT fits the tail portion of a probability distribution by a Generalized Pareto Distribution (GPD) with parameters. The GPD for high extreme value is defined as:

$$F(a) = P(A - th > a | A > th) \sim \left(1 + \frac{\gamma a}{\beta}\right)^{-\frac{1}{\gamma}}, \tag{14}$$

where th is the initial threshold of anomaly scores, γ and β are shape and scale parameters of GPD, a is a value of anomaly score. The portion below the threshold th is denoted as $A-th$, and the threshold th is empirically set to a high quantile. The final threshold z_q is then computed as:

$$z_q = th + \frac{\hat{\beta}}{\hat{\gamma}}\left(\left(\frac{qn}{N_{th}}\right)^{-\hat{\gamma}} - 1\right), \tag{15}$$

where q is the desired probability, n is the total number of anomaly scores, N_{th} is the number of peaks, i.e. the number of $a_t\ s.t.\ a_t > th$. The values of $\hat{\gamma}$ and $\hat{\beta}$ are estimated by Maximum Likelihood Estimation.

IV. Experiments

This section describes datasets followed by experimental settings and evaluation metrics. We conduct experiments to show the effectiveness of the proposed approach and justify our design choices.

A. Datasets

We evaluate DeconAnomaly on five datasets, i.e. SMAP[2] [10], MSL[2] [10], SMD[3] [12], PAMAP2[4] [24], and VAM[1]. They were collected from various systems. SMAP is from an environmental monitoring satellite named Soil Moisture Active Passive, MSL is from a rover designed for NASA's Mars

[2]github.com/khundman/telemanom
[3]github.com/NetManAIOps/OmniAnomaly
[4]archive.ics.uci.edu/ml/datasets/PAMAP2+Physical+Activity+Monitoring

Science Laboratory mission, SMD is from server machines, PAMAP2 is from people performing physical activities, and VAM is from a vinyl acetate monomer production plant simulator [25].

SMAP, MSL, SMD, and PAMAP2 have been used in past studies and are publicly available. SMAP, MSL, and SMD were collected for unsupervised anomaly detection, including a training set and a test set for each. We use the training and test sets but exclude T-10 data in SMAP based on comments by the data provider[5]. PAMAP2 is a generic dataset for multiple machine-learning tasks. We prepare a dataset for anomaly detection on PAMAP2 since it does not include the definition of its training set, its test set, normal samples, and abnormal samples. PAMAP2 contains data on 18 different physical activities performed by nine participants. We use data from eight participants, excluding those who performed few activities. The eight participants performed the same eight activities, and the data is regarded as normal data. The training set is data from seven out of the eight participants. The test set consists of normal data from the rest participant and abnormal data randomly sampled from the rest activities.

We generate VAM using a publicly available simulator[6]. The simulator includes a plant model, function blocks to simulate disturbance or failure, and a procedure for its start-up operations. The procedure includes five checkpoints forming five segments that appear in the same order at every execution. We collect normal data by running the simulator six times and maintaining the plant's normal status. The start time of the first operation after each checkpoint has been randomly changed to introduce variations in sensor readings. We also collect abnormal data activating a function block to inject a disturbance or failure. The function blocks simulate 14 categories of disturbances and 22 categories of failures. We select 15 malfunctions including five types of disturbances[7] and ten types of failures[8] for the experiments, and we allocate them so that each segment has the same number of malfunctions. We run the simulator for each disturbance or failure and activate it at the allocated segment. We collect additional normal data from the beginning of the allocated segment until the function block is activated and abnormal data for 30 min. in the simulation time after the function block is activated. The training set consists of the normal data for five runs without disturbance or failure injection. The test set consists of the rest normal data and the abnormal data.

We assign operating modes and sensor groups based on their side information. SMAP, MSL, and SMD were collected from different entities with compatible schemas. We assign a unique operating mode for each entity, supposing each has its own operating mode. As for PAMAP2, we assign a unique operating mode for each activity and a unique sensor group for each measurement unit, i.e., three inertial measurement units and a heart rate monitor. As for VAM, we assign a unique operating mode to each period segmented by checkpoints and a unique sensor group to each group of sensors shown on the same page of the piping and instrumentation diagram. The number of operating modes is 54, 27, 28, 8, and 5, for SMAP, MSL, SMD, PAMAP2, and VAM, respectively. The number of sensor groups is 1, 1, 1, 4, and 7 for SMAP, MSL, SMD, PAMAP2, and VAM, respectively.

[5]github.com/khundman/telemanom/issues/8

[6]www.omegasim.co.jp/contents_e/product/vm/trial

[7]MAL05, MAL06, MAL07, MAL13 and MAL14

[8]MAL15, MAL16, MAL17, MAL18, MAL19, MAL20, MAL21, MAL22, MAL26 and MAL27

TABLE I: F_1-score for each model. The best result is indicated with the bold characters.

Method \ Dataset	SMAP	MSL	SMD	PAMAP2	VAM	Average
OC-SVM	0.3814	0.0000	0.3524	0.9289	0.5615	0.4448
HMM	0.6090	0.5117	0.2306	0.9308	0.7931	0.6150
EncDec-AD	0.7031	0.7008	0.4362	0.9857	0.4272	0.6506
LSTM-AD	0.7580	0.7457	0.4519	0.9630	0.8401	0.7517
OmniAnomaly	**0.8102**	0.7533	0.5011	0.9311	0.4359	0.6863
DeconAnomaly	0.7259	**0.8857**	**0.7660**	**0.9979**	**0.8861**	**0.8523**

TABLE II: Precision for each model. The best result is indicated with the bold characters.

Method \ Dataset	SMAP	MSL	SMD	PAMAP2	VAM	Average
OC-SVM	0.5088	0.0000	0.2318	0.8672	0.4015	0.4019
HMM	0.4596	0.3708	0.1384	0.8706	0.6571	0.4993
EncDec-AD	0.6340	0.9387	0.2998	0.9718	0.2716	0.6232
LSTM-AD	0.7059	0.9227	0.3025	0.9287	0.7770	0.7274
OmniAnomaly	0.7664	0.7317	0.3528	0.8711	0.2861	0.6016
DeconAnomaly	**0.9560**	**0.9496**	**0.7301**	**0.9963**	**0.9531**	**0.9170**

TABLE III: Recall for each model. The best result is indicated with the bold characters.

Method \ Dataset	SMAP	MSL	SMD	PAMAP2	VAM	Average
OC-SVM	0.3050	0.0000	0.7354	**1.0000**	0.9333	0.5948
HMM	**0.9023**	0.8249	0.6887	**1.0000**	**1.0000**	**0.8832**
EncDec-AD	0.7890	0.5591	0.8002	**1.0000**	**1.0000**	0.8297
LSTM-AD	0.8183	0.6257	**0.8926**	**1.0000**	0.9143	0.8502
OmniAnomaly	0.8594	0.7764	0.8646	**1.0000**	0.9147	0.8830
DeconAnomaly	0.5851	**0.8298**	0.8056	0.9994	0.8280	0.8096

B. Experimental setting

DeconAnomaly has three stages for training models, i.e., Stage I, Stage Ⅱ, and Stage Ⅳ. In the experiments, models are trained with 500 epochs in total. The corresponding training steps are 211,000; 89,000; 110,500; 196,500; and 148,500; for SMAP, MSL, SMD, PAMAP2, and VAM, respectively. Training steps in Stage Ⅱ and Stage Ⅳ are set to 500 and 9,500, respectively. The rest training steps are used in Stage I.

We sample all possible territories under the following two conditions: 1) a territory should include an operating mode or contiguous operating modes, and 2) a territory should include the same sensor groups for each of its operating modes. The order of operating modes is determined for each dataset. As for PAMAP2 or VAM, it is the temporal order in the sequence. As for SMAP, MSL, or SMD, it is the dictionary order of its file names. The resulting number of territories is 1,485; 378;

TABLE IV: F_1-score for each multiple modeling strategy. The best result is indicated with the bold characters.

Method \ Dataset	SMAP	MSL	SMD	PAMAP2	VAM	Average
Individuals	0.7212	0.8074	0.5103	0.9924	0.4678	0.6998
Single	0.7372	0.5220	**0.8264**	0.8614	0.8882	0.7670
DeconAnomaly-c	**0.7881**	**0.8925**	0.7721	0.9965	**0.8973**	**0.8693**
DeconAnomaly	0.7259	0.8857	0.7660	**0.9979**	0.8861	0.8523
Improvement (%)	-7.89	-0.76	-0.79	0.14	-1.24	-1.95

406; 540; and 868; for SMAP, MSL, SMD, PAMAP2, and VAM, respectively.

Training data and validation data are sequentially sampled for each training set. The first 80% is used as training data, and the rest is used as validation data. They are normalized with min-max normalization.

We use the same parameter setting throughout the experiments unless stated. The input sequence length is 60, units in a hidden layer are 256, training epochs are 500, the batch size is 256, q is 0.001, and th is 0.95. We use Adam for training models.

Precision, recall, and F_1-score are used as evaluation metrics for the performance of DeconAnomaly and baseline models. The evaluation metrics are computed after the adjustment of prediction results as in the previous work [26].

C. Comparison with other methods

We compare DeconAnomaly with a variety of competing methods, including three neural network-based methods, namely EncDec-AD [11], LSTM-AD [9], OmniAnomaly [12], along with a conventional one-class learning method, One-Class SVM (OC-SVM) [27] and a conventional sequence learning method, Hidden Markov Models (HMM). EncDec-AD and LSTM-AD are chosen because of the similarity with the models of DeconAnomaly. OmniAnomaly is chosen as a baseline with a more sophisticated model architecture. The competing methods train a model for each operating mode and sensor group combination. The threshold for each model is determined by the Peaks-Over-Threshold approach on the validation data.

As shown in Table I, DeconAnomaly shows the best F_1-score in most cases and the best one on average. Although DeconAnomaly uses a simpler model architecture than OmniAnomaly, it outperforms OmniAnomaly. LSTM-AD outperforms EncDec-AD. This result is due to a way to compute the anomaly score rather than the difference between prediction and reconstruction as discussed in Sec. IV-F. Corresponding precision and recall are shown in Table. II and Table III. These tables show that the performance improvement by DeconAnomaly is attributed to improvement in precision rather than recall. Since a leaf model combines data in its territory and uses larger data at training than baseline models, DeconAnomaly more easily suppresses false positives.

D. Comparison with other multiple modeling strategies

In order to evaluate how much the optimal set exploration increases anomaly detection performance, we compare

TABLE V: Precision for each multiple modeling strategy. The best result is indicated with the bold characters.

Method \ Dataset	SMAP	MSL	SMD	PAMAP2	VAM	Average
Individuals	0.6353	0.9470	0.3709	0.9854	0.3053	0.6488
Single	**0.9634**	**0.9536**	**0.8745**	**0.9972**	**0.9578**	**0.9493**
DeconAnomaly-c	0.7655	0.9478	0.6967	0.9935	0.8805	0.8693
DeconAnomaly	0.9560	0.9496	0.7301	0.9963	0.9531	0.9170
Improvement (%)	24.89	0.19	4.79	0.28	8.25	5.49

TABLE VI: Recall for each multiple modeling strategy. The best result is indicated with the bold characters.

Method \ Dataset	SMAP	MSL	SMD	PAMAP2	VAM	Average
Individuals	**0.8339**	0.7037	0.8180	**0.9994**	**1.0000**	0.8710
Single	0.5971	0.3593	0.7833	0.7581	0.8280	0.6652
DeconAnomaly-c	0.8121	**0.8432**	**0.8657**	**0.9994**	0.9147	**0.8870**
DeconAnomaly	0.5851	0.8298	0.8056	**0.9994**	0.8280	0.8096
Improvement (%)	-27.95	-1.59	-6.95	0.00	-9.48	-8.73

DeconAnomaly with two naive solutions and a variant of DeconAnomaly. One naive solution, Single, trains a model for all the regions. The other naive solution, Individuals, trains a model for each region. The variant, DeconAnomaly-c, determines the optimal set of models without the penalty term. The model architecture is the same as in DeconAnomaly. Their F_1-scores are shown in Table IV. Improvement in the table is the relative improvement of DeconAnomaly from DeconAnomaly-c. DeconAnomaly-c obtains the best F_1-score on average, DeconAnomaly obtains the second best. DeconAnomaly-c and DeconAnomaly outperform Individuals for all the datasets. The relationships between F_1-score and the number of models are shown in Fig. 3. The worst result for each case is obtained with either the minimum or maximum number of models. These indicate that combining appropriate regions increases anomaly detection performance due to striking a balance between precision and recall. Their precision and recall are shown in Table V and VI, respectively. DeconAnomaly-c and DeconAnomaly are always better than Individuals in terms of precision. DeconAnomaly-c and DeconAnomaly are always better than Single in terms of recall, except for the case of SMAP by DeconAnomaly. Since DeconAnomaly-c and DeconAnomaly combine data of related regions at the training and test phases, they are robust for false positives but keep reasonable sensitivity for anomaly detection.

Although DeconAnomaly gives worse F_1-scores than DeconAnomaly-c, the difference is less than 2% for all datasets but SMAP, whereas DeconAnomaly achieves the results with a significantly smaller number of models. The number of models for each approach is shown in Table VII. Compression ratio is the ratio of the number of models with DeconAnomaly to that with DeconAnomaly-c. This result shows the effect of the penalty term. Raw compression ratio is the ratio of the number of models with DeconAnomaly-c to that with Individuals. This result shows the effect of maximizing the detection score. The overall compression ratio

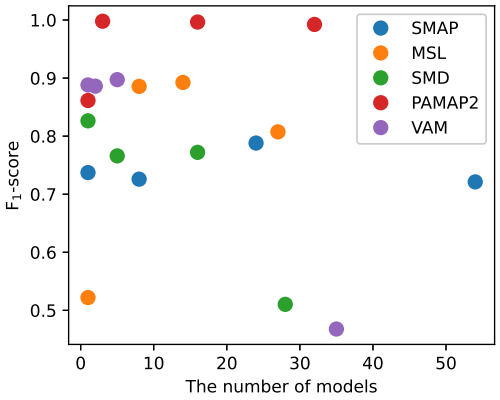

Fig. 3: Relationships between F_1-score and the number of models.

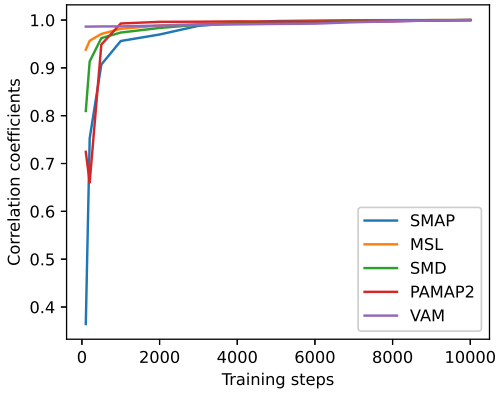

Fig. 4: Pearson correlation between observed detection scores and their final values.

TABLE VII: The number of models determined by each approaches and the compression ratio by DeconAnomaly.

Method \ Dataset	SMAP	MSL	SMD	PAMAP2	VAM	Average
Individuals	54	27	28	32	35	35.4
Single	1	1	1	1	1	1
DeconAnomaly-c	24	14	16	16	5	15
DeconAnomaly	8	8	5	3	2	5.2
Compression ratio (%)	33.33	57.14	31.25	18.75	40.00	36.10
Raw compression ratio (%)	44.44	51.85	57.14	50.00	14.29	42.61
Overall compression ratio (%)	14.81	29.63	17.86	9.38	5.71	15.48

is the ratio of the number of models with DeconAnomaly to that with Individuals. This result shows the overall effect of striking a balance between performance and complexity. The smaller those scores are, the better their reductions are. Regardless of with or without the penalty term for complexity, the number of models in the optimal set is smaller than the maximum number. These results show that not only the penalty term but also maximizing the detection score contribute to reducing the number of models. As a result, the number of models decreases by about 70%, at least from Individuals.

E. Order predictability

DeconAnomaly estimates the optimal set of models with smaller training steps at transfer learning. In order to make it feasible, the order of candidate models in detection scores needs to be predicted with a limited budget. We investigate if the detection scores with limited training steps are correlated with their final values, which is the best detection score with 10,000 training steps at fine-tuning. Pearson correlation between observed detection scores and their final values are shown in Fig. 4. After 500 training steps, the correlation coefficients become more than 0.8 for all datasets and 0.9 for all but the VAM dataset. This result suggests that the order can be predicted well with a limited budget.

F. Ablation study

In DeconAnomaly, leaf models fully utilize parameter values in the root model as many as possible, all their parameters are fine-tuned, and the root model is trained with a loss term to enhance transferability. We examine the effects of those design choices by comparing DeconAnomaly with its four variants, i.e. 'Baseline', 'FCL random initialization', 'Without transferability constraint', 'Only decoder', and 'Only FCL'. 'Baseline' is DeconAnomaly. In 'FCL random initialization', the gNet for each leaf model is randomly initialized. In 'Without transferability constraint', the root model is trained without the second term of Eq. (5). In 'Only decoder', only the decoder for each leaf model is fine-tuned. In 'Only FCL', only the gNet for each leaf model is fine-tuned. We introduce the winning rate for comparison. The winning rate is defined as the rate of the number of models whose root mean square error on validation data is improved by pre-training.

The VAM dataset is used for this study. We randomly sample 100 territories for evaluation and then train two models for each territory. They have the same architecture; however, one is initialized by parameters in the root model, and the other is randomly initialized before the training. The same training steps of 10,000 is set for each model, including the root model, to check if the negative transfer occurs.

The results are shown in Fig. 5. The winning rate decreases as the training steps increase because the effect of the negative transfer becomes apparent. 'Baseline' achieves the most successful transfer learning. 'FCL random initialization' shows a significantly low winning rate over training. This result indicates that parameter values in pre-trained gNets, which learn the relatedness of sensor groups, are crucial for successful transfer learning at DeconAnomaly. 'Without transferability constraint' shows a little lower winning rate than 'Baseline' and a decreasing trend over training. This means the transferability constraint has a modest but positive impact on transfer learning. The lowest winning rate with 10,000 training steps is given by 'Only FCL', followed in order by 'Only decoder' and 'Baseline'. The winning rate is correlated to the number of fine-tuned parameters. The more parameters in a model are fine-tuned, the easier the model adapts to its training data.

DeconAnomaly computes an anomaly score for a segment from reconstruction errors at the latest observation. We verify

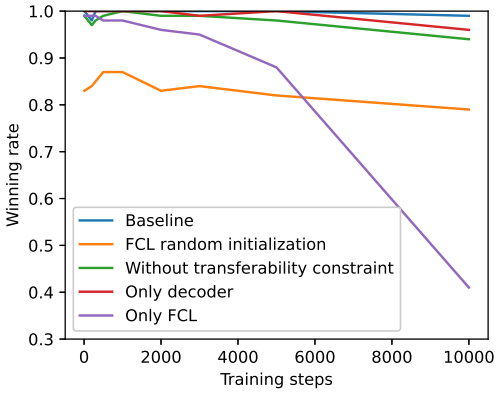

Fig. 5: Development in winning rate at transfer learning.

TABLE VIII: F_1-score based on each definition of anomaly score.

Method \ Dataset	SMAP	MSL	SMD	PAMAP2	VAM	Average
All	0.6197	0.1150	0.3410	0.8601	0.8001	0.5472
Latest	**0.7259**	**0.8857**	**0.7660**	**0.9979**	**0.8861**	**0.8523**

performance degradation due to taking all reconstruction errors into account. F_1-scores from both settings are shown in Table VIII. Here 'All' denotes the anomaly score computed with all the reconstruction errors, while 'Latest' denotes the anomaly score computed with the reconstruction errors of the latest observation. All the cases show significant degradation by considering the reconstruction errors.

V. Conclusion

We proposed DeconAnomaly, a deep learning framework to estimate the optimal set of models using transfer learning for unsupervised anomaly detection under a multiple modeling strategy. It utilizes transfer learning to reduce computational costs and estimate the optimal model set. The experimental results show clear advantages of the proposed framework, and the ablation study justifies our design choices.

References

[1] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[2] H. Ma, Y. Hu, and H. Shi, "A novel local neighborhood standardization strategy and its application in fault detection of multimode processes," *Chemometrics and Intelligent Laboratory Systems*, vol. 118, pp. 287–300, 2012.

[3] C. Tong, A. Palazoglu, and X. Yan, "An adaptive multimode process monitoring strategy based on mode clustering and mode unfolding," *Journal of Process Control*, vol. 23, no. 10, pp. 1497–1507, 2013.

[4] S. J. Zhao, J. Zhang, and Y. M. Xu, "Monitoring of processes with multiple operating modes through multiple principle component analysis models," *Industrial & Engineering Chemistry Research*, vol. 43, no. 22, pp. 7025–7035, 2004.

[5] S. Natarajan and R. Srinivasan, "Multi-model based process condition monitoring of offshore oil and gas production process," *Chemical Engineering Research and Design*, vol. 88, no. 5-6, pp. 572–591, 2010.

[6] S. Tan, F. Wang, J. Peng, Y. Chang, and S. Wang, "Multimode process monitoring based on mode identification," *Industrial & Engineering Chemistry Research*, vol. 51, no. 1, pp. 374–388, 2012.

[7] K. Zhang, K. Peng, and J. Dong, "A common and individual feature extraction-based multimode process monitoring method with application to the finishing mill process," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 4841–4850, 2018.

[8] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, Jul. 2009. [Online]. Available: https://doi.org/10.1145/1541880.1541882

[9] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long short term memory networks for anomaly detection in time series," in *ESANN 2015 23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2015, pp. 89–94.

[10] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," in *Proceedings of the 24th ACM SIGKDD int'l conf. on knowledge discovery & data mining*, 2018, pp. 387–395.

[11] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.

[12] 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 Int'l Conf. on Knowledge Discovery & Data Mining*, 2019, pp. 2828–2837.

[13] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder," *IEEE Rob. Autom. Lett.*, vol. 3, no. 3, pp. 1544–1551, 2018.

[14] D. Erhan, P.-A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent, "The difficulty of training deep architectures and the effect of unsupervised pre-training," in *Artificial Intelligence and Statistics*. PMLR, 2009, pp. 153–160.

[15] A. Sagheer and M. Kotb, "Unsupervised pre-training of a deep lstm-based stacked autoencoder for multivariate time series forecasting problems," *Scientific reports*, vol. 9, no. 1, pp. 1–16, 2019.

[16] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" *arXiv preprint arXiv:1411.1792*, 2014.

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conf. on computer vision and pattern recognition*, 2014, pp. 580–587.

[18] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Int'l conf. on artificial neural networks*. Springer, 2018, pp. 270–279.

[19] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese, "Taskonomy: Disentangling task transfer learning," in *Proceedings of the IEEE conf. on computer vision and pattern recognition*, 2018, pp. 3712–3722.

[20] K. Dwivedi and G. Roig, "Representation similarity analysis for efficient task taxonomy & transfer learning," in *Proceedings of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2019, pp. 12 387–12 396.

[21] J. Song, Y. Chen, X. Wang, C. Shen, and M. Song, "Deep model transferability from attribution maps," *arXiv preprint arXiv:1909.11902*, 2019.

[22] J. Yang, S. Rahardja, and P. Fränti, "Outlier detection: how to threshold outlier scores?" in *Proceedings of the int'l conf. on artificial intelligence, information processing and cloud computing*, 2019, pp. 1–6.

[23] A. Siffer, P.-A. Fouque, A. Termier, and C. Largouet, "Anomaly detection in streams with extreme value theory," in *Proceedings of the 23rd ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, 2017, pp. 1067–1075.

[24] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *2012 16th Int'l Symposium on Wearable Computers*. IEEE, 2012, pp. 108–109.

[25] Y. Machida, S. Ootakara, H. Seki, Y. Hashimoto, M. Kano, Y. Miyake, N. Anzai, M. Sawai, T. Katsuno, and T. Omata, "Vinyl acetate monomer (vam) plant model: a new benchmark problem for control and operation study," *IFAC-PapersOnLine*, vol. 49, no. 7, pp. 533–538, 2016.

[26] 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 Conf.*, 2018, pp. 187–196.

[27] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.