ANOMALY DETECTION WITH COMPLEX DATA STRUCTURES

A Dissertation

by

YUENING LI

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Xia Hu
Committee Members,	Yu Ding
	Theodora Chaspari
	Anxiao Jiang
Head of Department,	Scott Schaefer

December 2021

Major Subject: Computer Science

ABSTRACT

Identifying anomalies with complex patterns is different from the conventional anomaly detection problem. Firstly, for cross-modal anomaly detection problems, a large portion of data instances within a multi-modal context is often not anomalous when they are viewed separately in each modality, but they present abnormal patterns or behaviors when multiple sources of information are jointly considered and analyzed. Secondly, for the attribution network anomaly detection problem, the definition of anomaly becomes more complicated and obscure. Apart from anomalous nodes whose nodal attributes are rather different from the majority reference nodes from a global perspective, nodes with nodal attributes deviate remarkably from their communities are also considered to be anomalies. Thirdly, given a specific task with the different data structures, the process of building a suitable and high-quality deep learning-based outlier detection system still highly relies on human expertise and laboring trials. It is also necessary to automatically search the suitable outlier detection models for different tasks. In this dissertation, we made a series of contributions to enable advanced anomaly detection techniques for complex data structures and discussing how to automatically design anomaly detection frameworks for various data structures.

DEDICATION

To my mother, father, grandfather, and grandmother.

ACKNOWLEDGMENTS

Throughout the writing of this thesis, I have received a great deal of support and assistance. It is a genuine pleasure to express my deep sense of thanks and gratitude and appreciation to my advisor, Professor Xia Hu, for his patience, inspiration, and generous support that made this thesis possible. His professional guidance helped me to establish the recognition of the scientific research, and shaped me into a better researcher. I also would like to thank my dissertation committee members, Professor Yu Ding, Professor Theodora Chaspari, and Professor Anxiao Jiang, for their time, advice, and support.

Also, I would like to acknowledge my lab mates and collaborators at the DATA (Data Analytics at Texas A&M) Lab in the Department of Computer Science and Engineering, for all of the collaborations, suggestions and fun time.

Most importantly, I would like to thank my parents for giving me the opportunities and experiences that have made me who I am.

Finally, I would like to thank the Texas A&M University and all the funding agencies, including National Science Foundation and Defense Advanced Research Projects Agency.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor Xia Hu [advisor], Professor Anxiao Jiang, and Professor Theodora Chaspari of the Department of Computer Science and Engineering, and Professor Yu Ding of the Department of Industrial & Systems Engineering. Chapter II and III are conducted based on the discussion with Dr. Jundong Li from University of Virginia. Chapter IV and V are conducted based on the work majorly done while I was interning at NEC Labs America.

All work conducted for the dissertation was completed by the student independently.

Funding Sources

This work is, in part, supported by National Science Foundation (#IIS-1657196, #IIS-1750074, #CNS-1816497).The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

1.1 Motivation and Challenges

Anomalies refer to the data points, items, events, observations, or behaviors that are rare and different from the majority in a given dataset. Over the past few decades, various successful anomaly detection methods have been developed and widely applied in a number of high-impact domains, including fraud detection, cybersecurity, algorithmic trading, and medical treatment [1]. Identifying anomalies from a swarm of normal instances not only enables the detection and early alert of malicious behaviors but also helps enhance the performance of myriad downstream machine learning tasks.

Anomaly detection refers to the problem of finding instances in a dataset that do not conform to the expected patterns or behaviors of the majority [1]. The anomaly detection problem has a wide range of applications in many high-impact domains, such as fraud detection [2], intrusion detection [3], and healthcare monitoring and alert [4]. Conventional anomaly detection methods can be broadly categorized into classification-based methods [5, 6] and clustering-based methods [7]. The classification-based methods are often applied to learn a classifier from a set of labeled training data and then classify the test data into normal classes or anomalous classes [5, 6]. The clustering-based methods, on the other hand, often assume that anomalies belong to small and sparse clusters, while normal instances belong to large and dense clusters [7].

Conventional detection algorithms are mainly based on the assumption that instances are independent and identically distributed (i.i.d.) [8]. While in many real-world scenarios, the data we collected often comes from **different sources** or can be represented by different feature formats, forming the so-called complex data structure. Traditional methods lack the capability to model these complex data structures for anomaly detection. For instance, in synthetic ID detection, the target is to differentiate generated fake identities from the true identities [9] of users. In this task, each identity is associated with information from different modalities, such as ID photos, bank

transaction histories, and using online social behaviors. In other practical scenarios, instances are often explicitly or implicitly connected with each other, resulting in the so-called **attributed networks** [10]. Attributed networks are increasingly used to represent various real-world systems since the synergy between network structures and nodal attributes. For instance, in social media, users are not only affiliated with profile information and personalized posts as nodal attributes but also linked with each other by different social relations [11]. **Time-series** representation learning is a fundamental task for time-series analysis. While significant progress has been made to achieve accurate representations for downstream applications, the learned representations often lack interpretability and do not expose semantic meanings. In the meanwhile, given a specific task with different data structures, the process of building a suitable and high-quality deep learning-based outlier detection system still highly relies on human expertise and laboring trials. It is also necessary to **automatically search** the suitable outlier detection models for different tasks.

Identifying anomalies with complex patterns is different from the conventional anomaly detection problem. Firstly, for *cross-modal anomaly detection* problem, a large portion of data instances within a multi-modal context is often not anomalous when they are viewed separately in each individual modality, but they present abnormal patterns or behaviors when multiple sources of information are jointly considered and analyzed. For instance, in bank transaction records, each transaction is associated with different kinds of information like user profile, account summary, transaction times, and the user signature, which are naturally of different modalities. In this case, if the user profile is not consistent with other sources of information of the same user, it might cause a bank fraud associated with a major crime. In other words, these anomalies present inconsistent behaviors across different modalities [12], and the accurate detection of these anomalies has significant implications in practice. Secondly, for *attribution network anomaly detection* problem, the definition of anomaly becomes more complicated and obscure. Apart from anomalous nodes whose nodal attributes are rather different from the majority reference nodes from a global perspective, nodes with nodal attributes deviate remarkably from their communities are also considered to be anomalies [13]. The topological structures are within irregular or non-Euclidean

spaces [14], and traditional anomaly detection methods [15] could not be directly applied to attributed networks. Thirdly, complex data structure with temporal correlations makes latent codes hard to interpret. Time-series data usually contain temporal correlations, which cannot be directly captured and interpreted by traditional image-focused disentangle methods. While traditional sequential models, like LSTM or LSTM-VAE, could be used to model the temporal correlations, they neither provide interpretable predictions, as is often criticized, nor have an disentangle mechanism. Fourthly, the naturally imbalanced distribution and complex data structures have introduced new challenges in designing an *automated anomaly detection framework*. We often need to find a suitable definition of the anomaly and its corresponding objective function for a given real-world data. One typical way to define the anomalies is to estimate the relative density of each sample and declare instances that lie in a neighborhood with low density as anomalies [16]. Yet, these density-based techniques perform poorly if the data have regions of varying densities. Another way to define anomalies is through clustering. An instance will be classified as normal data if it is close to the existing clusters, while the anomalies are assumed to be far away from any existing clusters [17]. However, these clustering-based techniques will be less effective if the anomalies form significant clusters among themselves [1]. The proper definition of anomalies not only requires domain knowledge from researchers and experience from data scientists but also needs thorough and detailed raw data analysis efforts.

1.2 Contributions

To tackle the core challenges of complex data structures, we made four main contributions in this dissertation:

- To model multi-modality structures, the first contribution of this research dissertation is to systematically examine and define the cross-modal anomaly detection problem and analyze the limitations of existing efforts. Then we propose a novel cross-modal deep learning framework CMAD which could project data from different modalities into a comparable consensus feature space for anomaly detection when the original multi-modal data have dif-

ferent distributions.

- To capture topological relations, we propose a spectral convolution and deconvolution-based framework - SpecAE, to project the attributed network into a tailored space to detect global and community anomalies. SpecAE leverages Laplacian sharpening to amplify the distances between representations of anomalies and the ones of the majority.

- To understand sequential correlations, we explore the time-series representation learning task. We investigate to generate hierarchical semantic concepts as the interpretable and disentangled representations of time-series and thus incorporating sequential data to anomaly detection tasks.

- To automatically design anomaly detection frameworks for various data structures, we identify a novel and challenging problem (*i.e.*, automated anomaly detection) and propose a generic framework AutoOD. To the best of our knowledge, AutoOD describes the first attempt to incorporate AutoML with an anomaly detection task and one of the first to extend AutoML concepts into applications from data mining fields.

- Considering a large number of anomaly detection tasks in industrial applications and their similarities, we design a full-stack, end-to-end system for outlier detection. The end-to-end outlier detection system includes database operations and maintenance, the search process of automated outlier detection (including the search space and the search strategy design), and a visualization module which designs for users to understand the detection results better.

The first three contributions lie in modeling complex data structures for anomaly detection settings, while the remaining one focuses on automatically search an optimal anomaly detection framework for a given dataset with specific data characteristics.

1.3 Dissertation Overview

The remainder of this dissertation is organized as follows:

- **Chapter 2: Cross-Model Anomaly Detection.** In this chapter, we identify anomalies whose patterns are disparate across different modalities and we refer the studied problem as cross-modal anomaly detection.

- **Chapter 3: Attributed Network Anomaly Detection.** In this chapter, we design an anomaly detection approach which jointly model the nodal attributes and topological dependencies through graph convolution to enlarge the modeling capacity, which preserves the key information of the input nodal attributes and the topological relations without the restriction of homophily.

- **Chapter 4: Disentangled Representation on Time Series.** In this chapter, we explore multi-level disentanglement strategies by covering both individual latent factors and group semantic segments for time series.

- **Chapter 5: Automated Outlier Detection.** In this chapter, we explore how to design an automated outlier detection algorithm to find an optimal deep neural network model for a given dataset. We carefully design a search space specifically tailored to the automated outlier detection problem, covering architecture settings, outlier definitions, and the corresponding objective functions. We propose a curiosity-guided search strategy to overcome the curse of local optimality and stabilize search process.

- **Chapter 6: An End-to-end Automated Anomaly Detection System.** In this chapter, we present PyODDS, a full-stack, end-to-end system for outlier detection. PyODDS describes the first attempt to incorporate automated machine learning with outlier detection, and belongs to one of the first attempts to extend automated machine learning concepts into real-world data mining tasks.

- **Chapter 7: Conclusions and Future Research Opportunities.** We conclude the dissertation with a summary of contributions and discuss potential research directions to the results presented.

2. CROSS-MODAL ANOMALY DETECTION[1]

2.1 Motivation

Existing anomaly detection methods predominately focus on data from a single source. While in many real-world scenarios, the data we collected often comes from different sources or can be represented by different feature formats, forming the so-called multi-modal data. For instance, in synthetic ID detection, the target is to differentiate generated fake identities from the true identities [9] of users. In this task, each identity is associated with information from different modalities, such as ID photos, bank transaction histories, and user online social behaviors. Compared with the data that is collected from a single source, when different modalities are presented together, they often reveal complementary insights into the underlying application domains and can further lead to better learning performance. For example, recent research advances have shown that the cross-modal correlations among different modalities are valuable for many data mining tasks such as classification and clustering [18, 19, 20, 21, 22]. Hence, it motivates us to investigate whether the anomaly detection problem could be benefited from analyzing multi-modal data.

In this chapter, we focus on identifying anomalies whose patterns are disparate across different modalities and we refer the studied problem as *cross-modal anomaly detection*, which is different from the conventional anomaly detection problem. The major reason is that a large portion of data instances within a multi-modal context are often not anomalous when they are viewed separately in each individual modality, but they present abnormal patterns or behaviors when multiple sources of information are jointly considered and analyzed. For instance, in bank transaction records, each transaction is associated with different kinds of information like user profile, account summary, transaction times and the user signature, which are naturally of different modalities. In this case, if the user profile is not consistent with other sources of information of the same user, it might cause a bank fraud associated with a crucial crime. In other words, these anomalies present inconsis-

[1]This chapter is reprinted with permission from "Deep structured cross-modal anomaly detection", by Yuening Li, Ninghao Liu, Jundong Li, Mengnan Du, Xia Hu, 2019. Proceedings of the International Joint Conference on Neural Networks (IJCNN). Copyright 2019 by IEEE.

tent behaviors across different modalities [12], and the accurate detection of these anomalies has significant implications in practice.

However, cross-modal anomaly detection remains a challenging task. The main reason is that, to distinguish the cross-modal anomalies from other normal instances, we need to develop a principled framework to model the correlations between different modalities as the distributions of different modalities could be very complicated and may vary remarkably. Existing efforts either aim at maximizing the correlations between different modalities through linear mapping [18, 19], or try to extract features from different modalities with low-rank approximation techniques [23, 24]. In other words, the main focus of these methods is to embed various modalities into a consensus low-dimensional consensus feature space with a shallow learning model. In fact, the correlations might not be well captured in the low-dimensional consensus feature space by these shallow models due to their limited representation ability. In a nutshell, these shallow models cannot fully capture the nonlinear correlations among different modalities, which necessitates the investigation of the cross-modal anomaly detection problem with deep learning models.

2.2 Cross-Modal Anomaly Detection (`CMAD`) Framework

In this section, we introduce the proposed cross-modal anomaly detection framework `CMAD` in details. We begin with a formal definition of the cross-modal anomaly detection problem, and then propose a principled method to learn the consensus data representations across different modalities. After that, we discuss how to leverage deep structure to extract effective feature representations. Finally, we will introduce how to perform cross-modal anomaly detection with the built deep learning model.

2.2.1 Problem Definition

Given K different modalities, $\mathbf{M}_1$, $\mathbf{M}_2$,..., $\mathbf{M}_K$ are the corresponding feature matrices of these K modalities in C different classes, where $\mathbf{M}_k \in \mathbb{R}^{N\times d_k}$, and d_k is the feature dimensionality of the k^{th} modality. N is the number of instances in the dataset. The feature representation of the i^{th} instance is denoted as $\mathbf{M}^i$, where $i \in \{1,2,...,N\}$. $\mathbf{M}_k^i$ denotes the feature vector of the i^{th}

instance from modality $\mathbf{M}_k$. The class label of the i^{th} instance in modality $\mathbf{M}_k$ is denoted as $\mathbf{y}_k^i$, where $\mathbf{y}_k^i \in \{1,2,...,C\}$. Our task is to find the anomalous instances whose patterns (w.r.t. class labels) are inconsistent across different modalities. For the notation convenience and without loss of generality, we only include two modalities $\mathbf{M}_A$ and $\mathbf{M}_B$ when introducing the methodology, where A, $B \in \{1,2,...,K\}$ and $A \neq B$, but our method can be easily extended to consider any number of modalities.

Based on the aforementioned terminologies, we define the problem of *cross-modal anomaly detection* as follows:

Theorem 1. ***Cross-Modal Anomaly Detection****: Given a dataset $\mathcal{X}$ which contains multiple modalities as data sources, such as $\mathbf{M}_A$, $\mathbf{M}_B$, we define that the i^{th} instance in $\mathcal{X}$ is regarded as an anomaly when:*

$$F(\mathbf{M}_A^i, \mathbf{M}_B^i) < \epsilon, \tag{2.1}$$

where $F(.,.)$ is a function measuring the cross-modal similarity between a pair of observations, whereas in this definition, the observations belong to the same instance. The parameter ϵ is a pre-defined threshold value such that if the measured similarity is larger than or equals to ϵ, we regard that the i^{th} instance is normal across the two modalities; otherwise, the i^{th} instance is a cross-modal anomaly.

2.2.2 Data Fusion across Different Modalities

To detect the cross-modal anomalies, the fundamental prerequisite is to develop a principled way to fuse the information from different modalities. To achieve this target, a prevalent way is to learn the mapping function which could project instances from different modalities into a consensus feature space. In this way, it presents us a unified feature space in which we can measure the cross-modal similarity of the data instances, based on which the cross-modal anomaly detection can be easily carried out.

To this end, we transform the instances from two modalities A and B into a consensus feature space with two linear transformation matrices, denoted as $\mathbf{U} \in \mathbb{R}^{d_A \times r}$ and $\mathbf{V} \in \mathbb{R}^{d_B \times r}$, where r is

the dimensionality of the resultant unified feature space and its value is much smaller than d_A and d_B. After transformation, we obtain the following data representations for modalities A and B:

$$\mathbf{M}_{\tilde{A}} = \mathbf{U}^T \mathbf{M}_A, \tag{2.2}$$

$$\mathbf{M}_{\tilde{B}} = \mathbf{V}^T \mathbf{M}_B. \tag{2.3}$$

Given a pair of instances i and j, we employ the cosine similarity to measure their cross-modal similarity in the transformed feature space:

$$\begin{aligned} F(\mathbf{M}_A^i, \mathbf{M}_B^j) &= \cos(\mathbf{M}_{\tilde{A}}^i, \mathbf{M}_{\tilde{B}}^j) \\ &= \frac{\mathbf{M}_{\tilde{A}}^i \cdot \mathbf{M}_{\tilde{B}}^j}{||\mathbf{M}_{\tilde{A}}^i|| \cdot ||\mathbf{M}_{\tilde{B}}^j||} \\ &= \frac{\mathbf{U}^T \mathbf{M}_A^i \cdot \mathbf{V}^T \mathbf{M}_B^j}{||\mathbf{U}^T \mathbf{M}_A^i|| \cdot ||\mathbf{V}^T \mathbf{M}_B^j||}. \end{aligned} \tag{2.4}$$

To find the cross-modal anomalies, we need to ensure that for the pair of instances with consistent patterns across different modalities (i.e., $\mathbf{y}_A^i = \mathbf{y}_B^j$), their cross-modal similarity in the transformed feature space should be high. To this end, the loss function can be expressed as follows:

$$\min_{\mathbf{U},\mathbf{V}} \sum_{\{i,j\} \in \mathcal{S}} 1 - F(\mathbf{M}_A^i, \mathbf{M}_B^j), \tag{2.5}$$

where the set $\mathcal{S}$ contains the instance pairs $\{i, j\}$ whose cross-modal observations are consistent, i.e., $\mathbf{y}_A^i = \mathbf{y}_B^j$.

Also, we need to penalize the instance pairs with inconsistent patterns across different modalities. In particular, in the training process, we not only make use of the instance pairs with consistent patterns across different modalities, but also incorporate an additional set of negative samples by negative sampling [25]. Let $\mathcal{N} = \{(p, q)\}$ be the set of the negative samples where their cross-modal patterns are inconsistent (i.e., $\mathbf{y}_A^p \neq \mathbf{y}_B^q$), then to learn the mapping matrices, the loss function for encouraging dissimilarities of negative samples can be presented in the following for-

mat:

$$\min_{\mathbf{U},\mathbf{V}} \sum_{(p,q)\in\mathcal{N}} \max\left(0, F(\mathbf{M}_A^p, \mathbf{M}_B^q) - \gamma\right). \tag{2.6}$$

In the above formulation, the hyperparameter γ, as a margin value, controls the penalty degree of the inconsistent patterns. Specifically, if $F(\mathbf{M}_A^p, \mathbf{M}_B^q) > \gamma$, it implies that the instance pairs with inconsistent patterns across different modalities are regarded as similar and it is necessary to penalize it in the loss function.

By combining Eq. (2.5) and Eq. (2.6), it leads to the following objective function for the transformation matrices learning:

$$\min_{\mathbf{U},\mathbf{V}} \sum_{\{i,j\}\in\mathcal{S}} 1 - F(\mathbf{M}_A^i, \mathbf{M}_B^j) + \lambda_0 \sum_{(p,q)\in\mathcal{N}} \max\left(0, F(\mathbf{M}_A^p, \mathbf{M}_B^q) - \gamma\right) + \lambda_1(||\mathbf{U}||_F^2 + ||\mathbf{V}||_F^2). \tag{2.7}$$

The above loss function consists of three terms. The first term is to "pull" the projections of different modalities together if their cross-modal patterns (w.r.t. instance class labels) are consistent, while the second term is to "push" them further apart otherwise (as shown in Fig. 2.1(b)). The resultant unified feature space is shown in Fig. 2.1(c). The parameter λ_0 controls the influence of negative sampling. The third term is a regularization term, which controls the bias-variance trade-off in order to avoid the over-fitting, and λ_1 is the regularization hyper-parameter. The aforementioned objective function can be effectively solved with coordinate descent methods, in which the variables $\mathbf{U}$ and $\mathbf{V}$ are updated in an alternating way until the objective function converges to a local optimum.

2.2.3 Deep Structured Cross-Modal Anomaly Detection Framework - CMAD

Deep neural networks have been successfully applied as a powerful feature learning strategy for different types of single modality data, including text, image and audio data [26]. The main reason for the success can be attributed to the fact that deep neural networks are capable of learning nonlinear mappings by extracting high-level abstractions from the input raw features. Over the past few decades, many successful deep neural networks such as the deep Boltzmann machines [27], auto-encoders [26], and recurrent neural networks [28] have been widely applied and achieved the

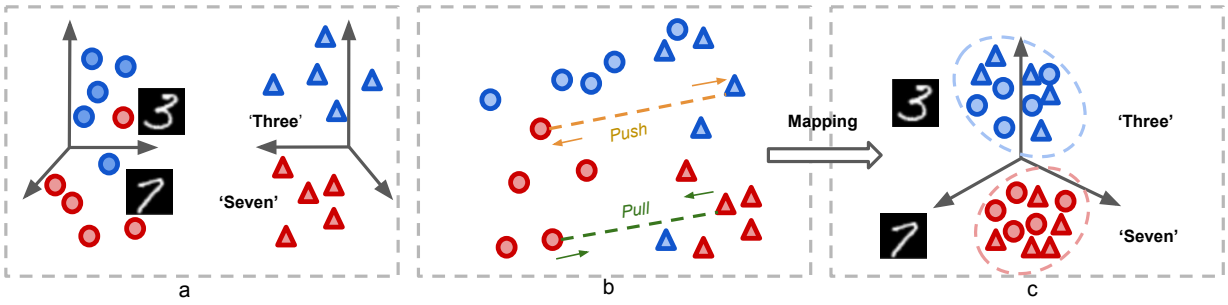


Figure 2.1: An illustration of the proposed cross-modal anomaly detection framework - CMAD.

state-of-the-art performance in many learning tasks.

Motivated by the success of deep neural networks, we develop a deep structured framework to characterize the features of different modalities for cross-modal anomaly detection. In particular, we employ a series of nonlinear mapping functions in the deep neural networks to map information of each modality into a consensus feature space, in which the instance pairs with consistent patterns across different modalities are pushed away while the pairs with inconsistent cross-modal patterns are pulled together. In this way, the trained model can help us identify the cross-modal anomalies in the test data. Compared with previously introduced linear projections, the developed method employs deep neural networks to learn nonlinear mappings. Thus it empowers a stronger ability to learn more effective feature representations from the original multi-modal data. In addition, the nonlinear mapping functions help to fully capture the nonlinear correlations among different modalities, which is otherwise difficult to characterize with conventional linear projection functions. It should be noted that the previously mentioned linear mapping is mathematically equivalent to imposing a single fully connected layer (without nonlinear activation functions) in the deep neural networks. Specifically, if we replace the linear mapping process (through mapping matrices $\mathbf{U}, \mathbf{V}$) with neural networks, then we have the loss function as:

$$\min_{\mathcal{W}} \sum_{\{i,j\}\in\mathcal{S}} 1 - F\left(f(\mathbf{M}_A^i), g(\mathbf{M}_B^j)\right) + \lambda \sum_{\{p,q\}\in\mathcal{N}} \max\left(0, F(f(\mathbf{M}_A^p), g(\mathbf{M}_B^q)) - \gamma\right), \tag{2.8}$$

where $f(\cdot)$ and $g(\cdot)$ denote neural network-like differentiable functions, $\mathcal{W}$ denotes the set of all

Algorithm 1 The training process of CMAD .

Input: Initialize a dataset $\mathcal{X}$, which contains multiple modalities as the data source, such as $\mathbf{M}_A, \mathbf{M}_B$
for each epoch **do**
  for each mini batch **do**
    for the positive training samples $\{i, j\}$ in $\mathcal{S}$ **do**
      $loss+ = 1 - F(\mathbf{M}_A^i, \mathbf{M}_B^j)$;
    end for
    for the negative training samples $\{p, q\}$ in $\mathcal{N}$ **do**
      $loss+ = \max(0, F(\mathbf{M}_A^p, \mathbf{M}_B^q) - \gamma)$;
    end for
  end for
  Update the model parameters with Adam;
end for
Output: The learned model parameters of CMAD .

Algorithm 2 Anomaly detection with CMAD .

Input: instances from two modalities, $\mathbf{M}_A, \mathbf{M}_B$
Output: the cross-modal anomalies
load the learned model parameters from Algorithm 1;
for $i \in \{1,2,...N\}$ **do**
  if $F(\mathbf{M}_A^i, \mathbf{M}_B^i) - \epsilon < 0$ **then**
    the i^{th} instance is a cross-modal anomaly;
  end if
end for

parameters of the deep neural network. We determine the architecture of the neural networks based on the data modalities to be handled. Specifically, we extract features from images through a convolutional neural network (CNN), and train a fully connected neural network to process text tags. The parameter λ controls the importance of the second term. In the experiments, we empirically set λ as 0.5 or 1.

Our goal is to minimize Eq. (2.8) through updating all the model parameters $\mathcal{W}$ in the deep neural networks. With an initialization of model parameters, the proposed deep model CMAD is optimized by Adam, which is an adaptive momentum based gradient descent method. The detailed are given in Algorithm 1. We make use of the dropout [29] in the training process to avoid

over-fitting. After the optimization process, the instances with consistent patterns across different modalities will be pulled together within a small distance, while the instances with inconsistent patterns across different modalities will be pushed away from each other. Finally, for example, the distribution of data instances in the projected latent space is shown as Fig. 2.1(c). Thus, we can use the CMAD to identify cross-modality anomalies by measuring the similarity across different modalities as shown in Algorithm 2.

In Fig. 2.1(a), different colors represent different classes of instances (e.g., the blue denotes the class of number 'three' while the red denotes the class of number 'seven') and each shape denotes one particular data modality (e.g., the circle denotes the image modality and the triangle denotes the text tag modality). Fig. 2.1(b) shows an example of processing inconsistent patterns (in orange dashed line) and consistent patterns (in green dashed line), where the goal is to find a consensus feature space in which consistent patterns across different modalities are pulled together while inconsistent patterns across different modalities are pushed away. Fig. 2.1(c) shows the final data distribution in the consensus feature space, and we can find that the instances with consistent patterns across different modalities are indeed grouped together.

2.3 Experiments

In this section, we conduct experiments to evaluate the effectiveness of our proposed CMAD framework in detecting cross-modal anomalies. We also perform a case study to visualize the distribution in the latent spaces to illuminate effectiveness of the CMAD .

2.3.1 Datasets

We use two real-world cross-modal datasets to assess the effectiveness of the proposed framework in cross-modal anomaly detection. The details of the datasets are as follow:

- **MNIST** [30]: In the MNIST dataset, we have 60,000 original images to represent ten different digits of $1\times28\times28$ pixels. After adding the text tag to each image and embed tags through different word embedding methods(Word2Vec [**?**], GloVe [**?**]) to 100 dimension vectors, we can synthetically generate the training data with two different modalities. The

Table 2.1: Hyaperparameters of the deep neural models.

Dataset	Dimension of the representations in the hidden layers
MNIST Images	784-1440-1280-320-150-50
MNIST Tags	100-100-50
RGB-D RGB Images	58575-1440-1280-320-150-50
RGB-D Depth Images	58575-1440-1280-320-150-50

image modality is fed into a CNN model, and the text modality is fed into a fully connected neural network for training.

- **RGB-D** [31]: In the RGB-D set, we have two modalities, RGB images and depth images to represent the same objects in the real-world. In our experiments, we include 960 RGB images and 960 corresponding depth images from five different kinds of objects in the RGB-D dataset, including apples, staplers, coffee-mugs, soda-cans, and toothpastes. Both modalities are fed into a CNN model for feature learning.

As there is no ground truth of anomalies in these datasets, thus we need to inject anomalies for evaluation. To generate the cross-modal anomalies, we adopt a widely used injection method - negative sampling, to create a number of instance pairs such that their patterns are inconsistent across different modalities. Originally, we only have instances whose patterns are consistent across different modalities (with the same class labels). To introduce the inconsistent instance pairs, we randomly sample a number of k instance pairs $\{p, q\}$ such that their cross-modal patterns are different such that $\mathbf{y}_A^p \neq \mathbf{y}_B^q$. To be more specific, in the testing stage, we inject 1,019 inconsistent pairs in the MNIST, and 545 inconsistent pairs in the RGB-D datasets. In the meanwhile, we have the same portion of instances with consistent behavior which have no overlap with the training stage.

2.3.2 Baseline Algorithms

We introduce several widely used baseline methods to compare the performance of cross-modal anomaly detection on multi-modal data:

- **CCA & KCCA** [18]. CCA models correlation across multiple modalities and implicitly maps instances into a lower-dimensional space, with the target to find the maximum correlations in the latent space by linear projections. Compared with CCA, KCCA [19, 32] uses an alternative projection strategy by nonlinearly mapping the multi-modal data into a consensus feature space with kernel tricks. We maximize the correlations of instances with consistent patterns in the training phase, and identify cross-modal anomalies by their cross-modal correlation in the test phase. Both CCA and KCCA are supervised.

- **PLS** [33]. PLS is also a supervised feature representation learning method based on linear transformation. The major difference between PLS and CCA is that PLS maximizes the covariance between different modalities through linear transformations. We identify the cross-modal anomalies in the same way as CCA and KCCA.

- **HOAD** [12]. HOAD is a supervised anomaly detection algorithm to find anomalies from the multi-modal data. It first obtains the spectral embeddings from the multi-modal data with an ensemble similarity matrix, and then calculates the anomaly score of each instance based on the cosine distance between different embeddings.

- **Embedding Network** [34, 35]. An supervised representation learning framework based on deep neural networks to extract features from different modalities, such as image and text. We distinguish the cross-modal anomalies by measuring the euclidean distances across different modalities after projecting instances into the latent space.

In our experiment, we evaluate the performance of cross-modal anomaly detection using the metrics of precision, recall, and accuracy.

Table 2.2: Performance on MNIST.

Method	Accuracy	Precision	Recall
CCA	0.6654	0.8911	0.6191
Kernel CCA(RBF)	0.6923	0.7062	0.6948
PLS	0.7235	0.6636	0.6233
HOAD	0.5118	0.9220	0.5115
Embedding Network	0.9504	0.9873	0.9127
CMAD	**0.9921**	**0.9971**	**0.9875**

We use TP, FP, TN, FN as the number of true positives, false positives, true negatives and false negatives, respectively, in the predicted results. Their definitions are listed as follows:

$$\text{precision} = \frac{TP}{TP + FP}, \tag{2.9}$$

$$\text{recall} = \frac{TP}{TP + FN}, \tag{2.10}$$

$$\text{accuracy} = \frac{TP + FP}{TP + TN + FP + FN}. \tag{2.11}$$

2.3.3 Anomaly Detection Performance

The parameter settings in our deep model `CMAD` vary among different modalities and datasets. Specifically, we use convolutional neural network (CNN) for the image modality [36, 37], and fully-connected neural network for the text tag modality. The hyperparameters of the feature dimensionality in each layer are listed in Table 2. Table 3 and Table 4 present the performance of cross-modal anomaly detection using different algorithms, which include the accuracy, precision and recall of the anomaly detection results.

As we can see from the tables, several observations are drawn as below.

First, `CMAD` outperforms the baseline methods CCA, KCCA, PLS and HOAD in most cases,

Table 2.3: Performance on RGB-D.

Method	Accuracy	Precision	Recall
CCA	0.7682	0.7171	0.7988
Kernel CCA(RBF)	0.8456	0.7134	0.9699
PLS	0.6580	0.8304	0.7539
HOAD	0.5137	**0.9345**	0.5124
Embedding Network	0.5138	0.5075	0.9346
CMAD	**0.9512**	0.9313	**0.9742**

which validates the importance of applying deep models for cross-modal anomaly detection by extracting more effective feature representations from the original multi-modal data.

Second, we compare CMAD against the Embedding Network method. Similar to our proposed CMAD , the Embedding Network is also a deep structure based approach. Based on the experimental results, both of the deep structured methods outperform other linear transformation based methods. These observations validate the limitation of the linear projection based methods, i.e., the linear transformation, as they cannot fully capture nonlinear correlations among different modalities for cross-modal anomaly detection.

There are two major differences between the proposed CMAD and the Embedding Network method. First, Embedding Network is developed to handle the text and image data only, while CMAD is able to handle different types of data with appropriate deep neural network designs. Second, the Embedding Network narrows the Euclidean distance of the pair of instances with consistent information across different modalities, whereas in CMAD , we not only pull samples with similar patterns across different modalities close to each other, but also push samples with dissimilar patterns across different modalities far away from each other. Thus, CMAD can learn better representations from the multi-modal data.

2.3.4 Hyperparameter Settings

There are two hyperparameters to be tuned in our method, i.e., the margin γ in Eq. (2.8) and the threshold ϵ in Eq. (2.2). In Fig. 2.2, we show how the performance of CMAD varies with different

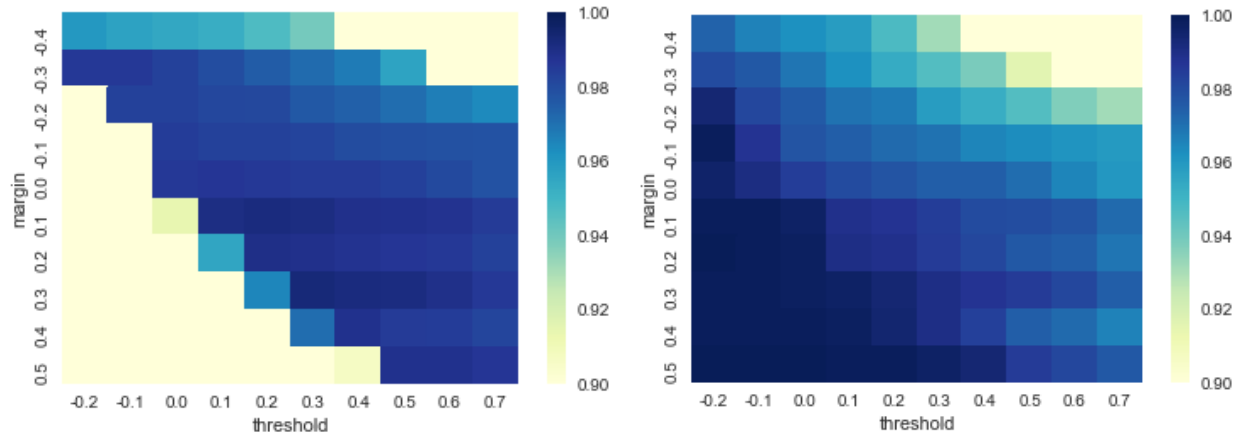


Figure 2.2: Precision and recall in the MNIST dataset with different hyperparameter settings.

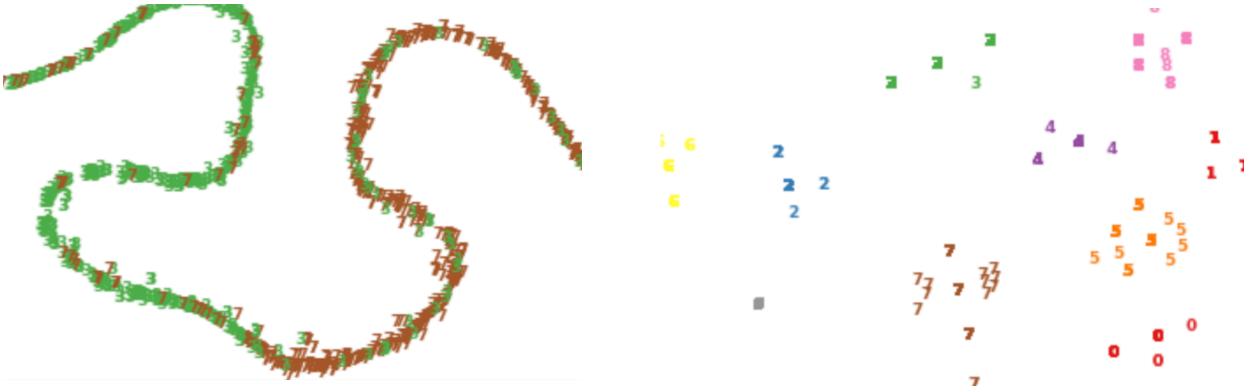


Figure 2.3: An illustration of the distribution of images and the distribution of tags.

hyperparameter settings. In particular, we tune the values of these hyperparameters within the range of {-0.4, -0.3, ... ,0,5} and {-0.2, -0.1, ... ,0,7}, respectively. In general, the algorithm performs better w.r.t. precision when the threshold ϵ is set to be small and the margin γ is set to be large. The accuracy achieves the best value when ϵ is 0.3 and γ equals to 0.3. In the meanwhile, we can find that the proposed `CMAD` could achieve a good performance (over 92%) in a wide range of hyperparameter settings, the loose condition give us more opportunity to find anomalies in the real-world applications.

Figure 2.4: Reconstruction result in MNIST.

2.3.5 Case Studies

Finally, we conduct two case studies to visualize the data distribution in the projected consensus feature space. Both of the cases are based on the MNIST dataset. The case studies are separated into two parts. The first part is to represent the data distribution of each individual modality in the latent space, and to demonstrate the effectiveness of the "pull" and "push" actions in our developed method. In the second part, we demonstrate the representation ability of `CMAD` across different modalities.

In the first case, in order to show that `CMAD` is capable to separate different instances in the projected consensus feature space, we visualize the data distribution in individual modalities after the training stage. To represent the data distribution, we first map the original instances into the projected feature space. Then we further reduce the dimensionality of the projected feature space to a two-dimensional visible feature space by PCA. As shown in Fig. 2.3, in each modality, the data instances which have similar patterns are separated from each other. In Fig. 2.3(left), we

can find that when the images with labels "three" and "seven" are taken as the test data, most of the images with label "three" (green in the figure) are separated from with the images with label "seven" (brown in the figure) in the projected two-dimensional space. In Fig. 2.3(right), we use the text tag as another data modality. We can find that the distributions of different tags could also be separated in the projected consensus feature space.

The second case is to demonstrate the representation ability of the developed method across different modalities. In other words, our target is to measure whether the distance between observations with similar modalities is closer than the distance between observations with dissimilar modalities after feature mapping. We extract one specific instance from one modality $\mathbf{M}_A$ in the projected consensus feature space and find its nearest (based on cosine distance) instances from the other modality $\mathbf{M}_B$.

As shown in Eq. (2.12), given an observation $\mathbf{M}_A^i$ where modality A denotes text data, we extract a set of semantically similar image observations from $\mathbf{M}_B$, and aggregate these observations to generate a new image:

$$\sum_{\forall M_B^j \in \mathbf{M_B}} \mathbb{I}\left(\frac{f(\mathbf{M}_A^i) \cdot g(\mathbf{M}_B^j)}{||f(\mathbf{M}_A^i)|| \cdot ||g(\mathbf{M}_B^j)||} - \epsilon > 0\right) \cdot \mathbf{M}_B^j. \tag{2.12}$$

After normalizing the generated image from Eq. (2.12) in the range of [0, 255] for visualization in the gray scale, we generate a new image which can represent the estimation of the distribution close to the given text $\mathbf{M}_A^i$.

According to finding the closest instances from other modalities, we could demonstrate that, the distances between the same objects from different modalities are closer than those denoting different objects in the latent space. The upper part are the selected images as the ground truth, and the lower part are reconstructed images through finding the closest neighbors which originally distributed in the text tag space.

We extract instances from the text modality to find the instances from the image modality with the closest cosine distances in the latent space, and reconstruct images from the text features. After

normalizing the nearest images, we represent them as new, reconstructed images in the gray scale. The reconstructed results are reported in Fig. **??**. The upper part contains the ground-truth images, and the lower part denotes to the reconstructed results. Comparing the reconstructed results with the original images, we can intuitively find that our algorithm is effective in pulling images with the consistent instances across different modalities together, and pushing those with inconsistent instances apart away.

3. ATTRIBUTED NETWORK ANOMALY DETECTION[1]

3.1 Motivation

Anomaly detection targets at identifying the rare instances that behave significantly different from the majority instances. Conventional detection algorithms are mainly based on the assumption that instances are independent and identically distributed (i.i.d.) [8]. But in many practical scenarios, instances are often explicitly or implicitly connected with each other, resulting in the so-called *attributed networks* [10]. Attributed networks are increasingly used to represent various real-world systems since the synergy between network structures and nodal attributes. For instance, in social media, users are not only affiliated with profile information and personalized posts as nodal attributes, but also linked with each other by different social relations [11]. Detecting anomalies in attributed networks have witnessed a surge of research interests from various domains, such as suspicious account detection in social media, abuse monitoring in healthcare systems [38], financial fraud monitoring [39], and network intrusion detection in cyber security [40].

However, the unique data characteristics of attributed networks bring several challenges to anomaly detection. Firstly, the definition of anomaly becomes more complicated and obscure. Apart from anomalous nodes whose nodal attributes are rather different from the majority reference nodes from a global perspective, nodes with nodal attributes deviate remarkably from their communities are also considered to be anomalies [13]. Secondly, the topological structures are within irregular or non-Euclidean spaces [14], and traditional anomaly detection methods [15] could not be directly applied to attributed networks. Thirdly, networks and nodal attributes are heterogeneous, and effective anomaly detection methods need to fuse these two data modalities more synergistically.

In this chapter, we aim to design an anomaly detection approach which jointly model the nodal attributes and topological dependencies through graph convolution to enlarge the modeling ca-

[1]This chapter is reprinted with permission from "Specae: Spectral autoencoder for anomaly detection in attributed networks", by Yuening Li, Xiao Huang, Jundong Li, Mengnan Du, Na Zou, 2019. Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM). Copyright 2019 by ACM.

pacity, which preserves the key information of the input nodal attributes and the topological relations without the restriction of homophily. Specifically, we introduce a graph deconvolution as a complementary process of graph convolution, which reconstruct the nodal attributes based on the topological dependencies. Previously, to handle the topological structures and the heterogeneity challenge, in network analysis, a widely-used and effective approach is to embed all information in attributed network into unified low-dimensional node representations. However, it would achieve suboptimal performance when directly applied to the anomaly detection. To achieve the join embedding, most existing network embedding algorithms [10] and recent joint embedding based anomaly detection methods [11] mainly rely on the homophily hypothesis, which implies that connected nodes tend to have similar nodal attributes [41]. Based on this assumption, topological relations could be introduced via involving regularization terms to minimize the distance between embedding representations of linked nodes [42]. Another effective way to encode networks is to employ graph convolutional networks (GCN) [43]. It could be considered as a special form of Laplacian smoothing that learns nodes' representations from their neighbors [44]. The homophily hypothesis or smoothing operations are not in line with anomaly detection. They might over-smooth the node representations, and make anomalous nodes less distinguishable from the majority within the community. For example, malicious users would have completely different nodal attributes than their friends. Customers who have written fake reviews might purchase the same products as the normal ones. Thus, existing joint learning models in network analysis could not be directly applied to anomaly detection.

3.2 Problem Statement

We use bold upper case letters (e.g., $\mathbf{X}$) to denote matrices, bold lowercase characters for vectors (e.g., $\mathbf{x}$). We denote the a^{th} row of the matrix $\mathbf{X}$ as $\mathbf{X}_{a,:}$, the b^{th} entry of the vector $\mathbf{x}$ as $\mathbf{x}_b$. Let $\mathcal{G} = \{\mathbf{A}, \mathbf{X}\}$ be an input attributed network, with n nodes interconnected by a network. $\mathbf{A} \in \mathbb{R}^{n \times n}$ denotes the corresponding adjacency matrix. Each node is associated with a set of m-dimensional nodal attributes. We collect all these nodal attribute as $\mathbf{X} \in \mathbb{R}^{n \times m}$.

Theorem 2. ***G****lobal Anomaly refers to a node whose nodal attributes are rare and significantly*

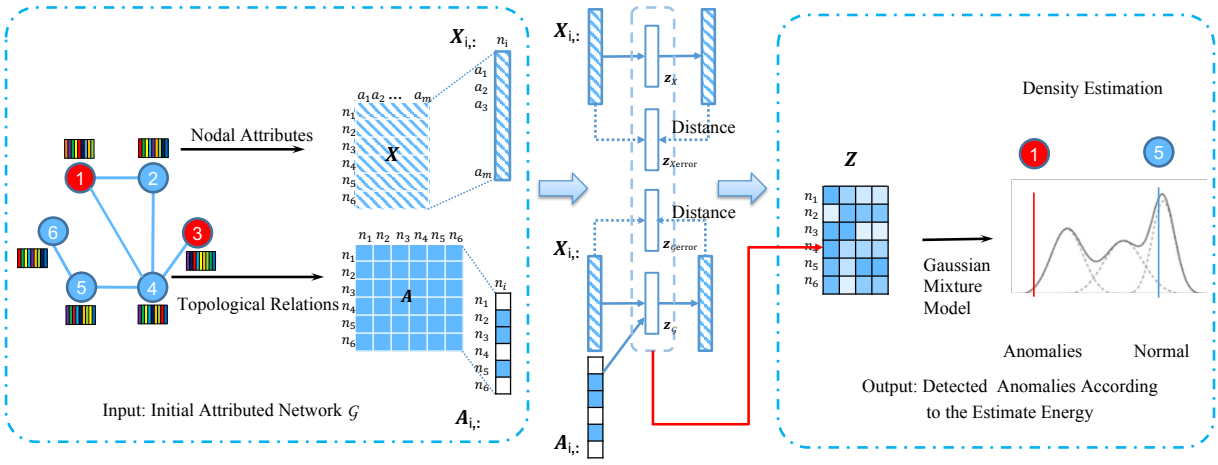


Figure 3.1: A spectral autoencoder based anomaly detection approach for attributed networks framework.

different from the nodal attributes of majority nodes, from a global perspective.

Theorem 3. ***Community Anomaly*** *is defined as a node i with nodal attributes that significantly deviate from the node i's neighbors' nodal attributes, based on the topological network structure.*

We define two types of anomalies. Detecting them in attributed networks is crucial. For instance, in social networks, the malicious, zombie, spam, or suspicious users' behaviors might deviate with most of the normal users'. We treat this type of users as global anomalies. There also could be a small portion of accounts with contents significantly different from their community, such as advertising/marketing users in college student clubs. We treat this type of users as community anomalies.

Given the aforementioned definitions, we formally define the anomaly detection in attributed networks problem as follows. Given a set of nodes $\mathcal{V}$ connected by an attribute network $\mathcal{G}$, we target at identifying the global anomalies and community anomalies in $\mathcal{V}$.

3.3 Spectral Autoencoder Framework

We propose a Spectral autoencoder based anomaly detection framework - SpecAE, for attributed networks. The pipeline of SpecAE is illustrated in Fig.1. Given an attributed network $\mathcal{G}$,

we leverage a tailored Spectral autoencoder to jointly embed nodal attributes and relations into a carefully-designed space $\mathbf{Z} = [\mathbf{Z}_X; \mathbf{Z}_{X\text{error}}; \mathbf{Z}_{\mathcal{G}}; \mathbf{Z}_{\mathcal{G}\text{error}}]$, which consists of four sources of components. To detect the global anomalies, we apply an autoencoder to all nodal attributes $\mathbf{X}$ to learn embedding representations $\mathbf{Z}_X$ as well as the reconstruction errors $\mathbf{Z}_{X\text{error}}$. In such a way, we could globally compare all nodes' attributes. To detect the community anomalies, we design novel graph convolutional encoder and deconvolutional decoder networks to learn nodes' community representations $\mathbf{Z}_{\mathcal{G}}$, based on each node's neighbors. The corresponding reconstruction errors are denoted as $\mathbf{Z}_{\mathcal{G}\text{error}}$. Later on, based on the tailored joint representations $\mathbf{Z}$, we estimate the suspicious level of each node u by measuring its embedding representation $\mathbf{Z}_{i,:}$'s energy in the Gaussian Mixture Model.

3.3.1 Tailored Embedding Spaces for Global Anomalies and Community Anomalies

Attributed networks bring both opportunities and challenges to the anomaly detection task. The two sources $\mathbf{A}$ and $\mathbf{X}$ bring more information than uni-modality, but make the definition of anomaly more complex.

To perform anomaly detection in $\mathcal{G}$, an intuitive approach is to apply the attributed network embedding [42] algorithm to project $\mathcal{G}$ into unified node embedding representations, and estimate the suspicious levels of nodes based on the learned representations. The goal of network embedding is to preserve nodes' major characteristics and map all nodes into a unified space, based on the homophily hypothesis [41] that nodes with similar topological structures tend to have similar node attributes. However, anomaly detection focuses on discriminating nodes with characteristics that are different from the majority, including nodes that are inconsistent with the homophily hypothesis. Thus, the conventional attributed network embedding methods will lead to a suboptimal result.

In this subsection, we propose to category anomalies in attributed networks into two classes, i.e., global anomalies and community anomalies. To distinguish these anomalies, we project $\mathcal{G}$ into two types of tailored low-dimensional spaces.

First, based on the definition of global anomaly, we employ an autoencoder to embed nodal

attributes $\mathbf{X}$ to learn the first type of tailored representations, i.e., $\mathbf{Z}_X$ and $\mathbf{Z}_{X\text{error}}$. The encoding and decoding processes are achieved via:

$$\begin{aligned} \mathbf{Z}_X &= \sigma\left(\mathbf{b}_e + \mathbf{X}\mathbf{W}_e\right), \\ \hat{\mathbf{X}} &= \sigma\left(\mathbf{b}_d + \mathbf{Z}_X\mathbf{W}_d\right), \end{aligned} \tag{3.1}$$

where $\hat{\mathbf{X}}$ denotes the reconstructed nodal attributes. Since anomalies are harder to reconstruct than normal nodes [45], we also include the reconstruction errors $\mathbf{Z}_{X\text{error}}$.

$$\mathbf{Z}_{X\text{error}} = \text{dis}(\mathbf{X}, \hat{\mathbf{X}}), \tag{3.2}$$

where the operation $\text{dis}(\cdot, \cdot)$ denotes a series of the distance measuring matrices such as the Euclidean distance and the cosine distance. In such a way, global anomalies would tend to have their representations in $\mathbf{Z}_X$ being different from the majority, as well as $\mathbf{Z}_{X\text{error}}$ being large.

Second, to identify community anomalies, we need to jointly consider $\mathbf{A}$ and $\mathbf{X}$. Based on their topological dependencies, we develop a novel graph convolutional encoder and graph deconvolutional decoder to learn the second type of tailored representations, i.e., $\mathbf{Z}_{\mathcal{G}}$ and $\mathbf{Z}_{\mathcal{G}\text{error}}$. Details are introduced as follows.

3.3.2 Graph Convolution and Deconvolution

Our goal is to learn nodes' community representations $\mathbf{Z}_{\mathcal{G}}$, which describe the expected behaviors of nodes according to their neighbors.

A straightforward solution is to apply GCN to embed $\mathcal{G}$. From a formulation perspective [44], GCN could be treated as a special form of *Laplacian Smoothing*. The *Laplacian smoothing* on each channel of the input features can be written as:

$$\mathbf{Y} = (\mathbf{I} - \gamma\tilde{\mathbf{D}}^{-1}(\tilde{\mathbf{D}} - \tilde{\mathbf{A}}))\mathbf{X}, \tag{3.3}$$

where $0 < \gamma \leq 1$ is a parameter which controls the weighting between the features of the current

node and the features of its neighbors, with a normalization trick as:

$$\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}} \rightarrow \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}, \tag{3.4}$$

where $\tilde{\mathbf{A}} := \mathbf{A} + \mathbf{I}$ and $\tilde{\mathbf{D}} := \sum_j \tilde{\mathbf{A}}_{ij}$.

Based on the definition of convolution in graph signals on the spectral domain, to generate a new matrix $\mathbf{Y}$ from $\mathbf{X}$ by applying the graph convolution as spectral filters, we have:

$$\mathbf{Y} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}, \tag{3.5}$$

which is a special form of *Laplacian Smoothing* when $\gamma = 1$ after replacing the *normalized graph Laplacian* $\mathbf{L} := \tilde{\mathbf{D}}^{-1}\tilde{\mathbf{A}}$ with the *symmetric normalized graph Laplacian* $\mathbf{L} := \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$.

However, GCN does not contain nodal attributes reconstruction procedure, which is useful in the anomaly detection task with three reasons. i) Training transformation functions base on one-class observations (only normal instances available) without the reconstruction error may easily lead the objective function converges to a local optimal when all of the node representations collapse into a small area (one point in extreme cases) in the latent space. ii) Repeatedly applying *Laplacian smoothing* might cause the nodal attributes over-mixed with their neighbors and make them indistinguishable, since it will be more difficult to identify each individual instance after the smoothness operation which has weakened their uniqueness of the original attributes. iii) Reconstruction error usually contains useful information as indicators for anomaly detection. For example, in [15] [46] [47] [45], the reconstruction error could be directly used as an anomaly score to rank the anomalous degree of nodes.

Thus, we decide to design the graph decoding (graph deconvolution) from the smoothed features as a complementary inverse process of graph convolution. Inspired by the digital images field, *sharpening* is an inverse process of blurring/smoothing [48]. Comparing with the smoothing process, which is done in the spatial domain by pixel averaging in neighbors, sharpening is to find the difference by the neighbors, done by spatial differentiation. A Laplacian operator is to restore

fine details of an image which has been smoothed or blurred. After Laplacian sharpening, we can reconstruct the nodal attributes from the fusion features caused by the graph convolution process.

If we replace the original GCN function in Eq. (3.5) as a general *Laplacian smoothing*, we will have:

$$\begin{aligned} \mathbf{Y} &= \mathbf{X} - \alpha(\mathbf{X} - \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}) \\ &= (1-\alpha)\mathbf{X} + \alpha\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}. \end{aligned} \tag{3.6}$$

Generalizing the above definition of graph convolution with adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ and the nodal attributes $\mathbf{X} \in \mathbb{R}^{n\times m}$ of n instances and m input channels, the propagation rule of the convolution layer can be written as:

$$\text{Conv}(\mathbf{X}, \mathbf{A}) = \sigma\left((1-\alpha)\mathbf{X} + \alpha\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}\right)\mathbf{W}_f, \tag{3.7}$$

where $\mathbf{W}_f$ is the trainable weight matrix in the convolution layer, σ is the activation function, e.g., $ReLU(\cdot)$. The parameter α is to control the weighting between the features of the current nodal attributes and the features of its neighbors. When $\alpha = 1$, we can treat the Eq. (3.7) as a FC layer in a MLP framework; if $\alpha = 0$, we have one graph convolutional layer. We conduct experiments under different parameter settings in the ablation studies section to illuminate the effect of the α.

Contrary to *Laplacian smoothing*, we compute the new features of nodal attributes through sharpening the features with their neighbors in order to reconstruct the features from the smoothed results. To magnify the difference between the current node and its neighbors, we will have:

$$\begin{aligned} \mathbf{Y} &= \mathbf{X} + \alpha(\mathbf{X} - \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}) \\ &= (1+\alpha)\mathbf{X} - \alpha\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}. \end{aligned} \tag{3.8}$$

Given the above definition of deconvolution to an attributed network $\mathcal{G}$, with adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ and the nodal attributes $\mathbf{Z} \in \mathbb{R}^{n\times m}$ of n instances and m input channels which after the

convolution process, the propagation rule of the deconvolution layer is:

$$\text{Deconv}(\mathbf{Z}, \mathbf{A}) = \sigma\left((1+\alpha)\mathbf{Z} - \alpha\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{Z}\right)\mathbf{W}_g, \tag{3.9}$$

where $\mathbf{W}_g$ is the trainable weight matrix in the deconvolution layer. After the sharpening process, we can reconstruct the original attributes from the smoothed features.

Given nodal attributes $\mathbf{X}$ with adjacency matrix $\mathbf{A}$, the compression network computes their low-dimensional representations $\mathbf{Z}$ as follows:

$$\begin{aligned} f(\mathbf{Z}_{\mathcal{G}}|\mathbf{X}, \mathbf{A}) &= \prod_{i=1}^{n} f(\mathbf{Z}_{\mathcal{G}_{i,:}}|\mathbf{X}_{i,:}, \mathbf{A}), \\ \text{with} \quad f(\mathbf{Z}_{\mathcal{G}_{i,:}}|\mathbf{X}, \mathbf{A}) &= \mathcal{N}(\mathbf{Z}_{\mathcal{G}_{i,:}}|\mu_{i,:}, \text{diag}(\sigma^2_{i,:})); \end{aligned} \tag{3.10}$$

$$\begin{aligned} g(\widetilde{\mathbf{X}}|\mathbf{Z}_{\mathcal{G}}, \mathbf{A}) &= \prod_{i=1}^{n} g(\widetilde{\mathbf{X}}_{i,:}|\mathbf{Z}_{\mathcal{G}_{i,:}}, \mathbf{A}), \\ \text{with} \quad g(\widetilde{\mathbf{X}}_{i,:}|\mathbf{Z}_{\mathcal{G}_{i,:}}, \mathbf{A}) &= \text{Deconv}(\mathbf{Z}_{\mathcal{G}_{i,:}}, \mathbf{A}); \end{aligned} \tag{3.11}$$

$$\mathbf{Z}_{\mathcal{G}\text{error}} = \text{dis}(\mathbf{X}, \widetilde{\mathbf{X}}), \tag{3.12}$$

$$\mathbf{Z} = [\mathbf{Z}_X; \mathbf{Z}_{\mathcal{G}}; \mathbf{Z}_{X\text{error}}; \mathbf{Z}_{\mathcal{G}\text{error}}], \tag{3.13}$$

where the $\mu_{i,:} := \text{Conv}_{\mu}(\mathbf{X}, \mathbf{A})$ denote the mean vector in Eq. (3.8), similarly, $\log\sigma_{i,:} := \text{Conv}_{\sigma}(\mathbf{X}, \mathbf{A})$. The latent variable $\mathbf{Z}$ of each node is shown in Eq. (3.13), which concatenates from four components in Eq. (3.1), Eq. (3.2), Eq. (3.10), and Eq. (3.12). $\mathbf{Z}_{\mathcal{G}}$ is learned by the graph convolutional encoder which including the neighbor attribution with the topological relations. $\mathbf{Z}_{\mathcal{G}\text{error}}$ is the reconstruction of the nodal attributes $\mathbf{X}$ base on the fusion representation $\mathbf{Z}_{\mathcal{G}}$ and the adjacency matrix $\mathbf{A}$.

3.3.3 Anomaly Detection via Density Estimation

Given the learned embedding representation $\mathbf{Z}$ of N samples and their soft mixture-component membership prediction $\hat{\gamma}$ (estimated from a softmax layer based on the low-dimensional repre-

sentation $\mathbf{Z}$), where $\hat{\gamma}$ is a K-dimensional vector and K is the number of mixture components in Gaussian Mixture Model (GMM), we can estimate the mixture probability $\hat{\phi}_k$, the mean value $\hat{\mu}_k$, the covariance vector $\hat{\Sigma}_k$ for component k in GMM respectively [16].

The sample energy can be inferred by:

$$E(\mathbf{z}) = -\log\left(\sum_{k=1}^{K}\hat{\phi}_k\frac{\exp -\frac{1}{2}(\mathbf{z}-\hat{\mu}_k)^T\hat{\Sigma}_k^{-1}(\mathbf{z}-\hat{\mu}_k)}{\sqrt{|2\pi\hat{\Sigma}_k|}}\right), \tag{3.14}$$

where we estimate the parameters in GMM as follows:

$$\hat{\gamma} = \text{softmax}(\mathbf{z}), \tag{3.15}$$

$$\hat{\phi}_k = \sum_{i=1}^{N}\frac{\hat{\gamma}_{ik}}{N}, \tag{3.16}$$

$$\hat{\mu}_k = \frac{\sum_{i=1}^{N}\hat{\gamma}_{ik}\mathbf{z}_i}{\sum_{i=1}^{N}\hat{\gamma}_{ik}}, \tag{3.17}$$

$$\hat{\mathbf{\Sigma}}_k = \frac{\sum_{i=1}^{N}\hat{\gamma}_{ik}(\mathbf{z}_i-\hat{\mu}_k)(\mathbf{z}_i-\hat{\mu}_k)^T}{\sum_{i=1}^{N}\hat{\gamma}_{ik}}. \tag{3.18}$$

3.3.4 Objective Function

Given an input attributed network $\mathcal{G} = \{\mathbf{A}, \mathbf{X}\}$, the objective function is constructed as follow:

$$\begin{aligned} J(\mathbf{W}) =& \mathbb{E}[\text{dis}(\mathbf{X}, \hat{\mathbf{X}})] + \mathbb{E}[\text{dis}(\mathbf{X}, \widetilde{\mathbf{X}})] + \lambda_1\mathbb{E}(E(\mathbf{Z})) + \lambda_2\sum_{i=1}^{K}\sum_{j=1}^{d}\frac{1}{\hat{\mathbf{\Sigma}}_{ij}} - \mathcal{KL}[f(\mathbf{Z}_\mathcal{G}|\mathbf{X},\mathbf{A})||g(\mathbf{Z}_\mathcal{G})] \\ &+\mathbb{E}_{f(\mathbf{Z}_\mathcal{G}|\mathbf{X},\mathbf{A})}\log(g(\widetilde{\mathbf{X}}|\mathbf{Z}_\mathcal{G},\mathbf{A})). \end{aligned} \tag{3.19}$$

The objective function includes four components:

- $\mathbb{E}[\text{dis}(\mathbf{X}, \hat{\mathbf{X}})], \mathbb{E}[\text{dis}(\mathbf{X}, \widetilde{\mathbf{X}})]$ are the loss function that characterizes the reconstruction error.
- $E(\mathbf{Z})$ denotes the sample energy of the GMM estimation in Eq. (3.14). Through minimizing the sample energy, we maximize the likelihood of non-anomalous samples, and predict

samples with top-K high energy as anomalies.

- To avoid trivial solutions when the diagonal entries of covariance matrices degenerate to 0, we penalize small values by the third component as a regularizer.

- We optimize the variational lower bound as the last two terms. $\mathcal{KL}[f(\cdot)||g(\cdot)]$ is the Kullback-Leibler divergence between $f(\cdot)$ and $g(\cdot)$. $g(\cdot)$ is a Gaussian prior $g(\mathbf{Z}_{\mathcal{G}}) = \prod_{i=1}^{n} g(\mathbf{Z}_{\mathcal{G}_{i,:}}) = \mathcal{N}(\mathbf{Z}_{\mathcal{G}_{i,:}}|0, \mathbf{I})$.

After optimizing the objective function, our proposed approach can be applied to detect anomalous to the input attributed network. For our testing data, we can rank the anomalous degree of each node according to the corresponding estimation energy in Eq. (3.14). According to the ranked energy, nodes with larger scores are more likely to be considered as anomalies.

3.4 Experiments

In this section, we evaluate the performance of our model with experiments on real-world datasets, to verify the effectiveness of the proposed approach.

3.4.1 Datasets

We adopt two real-world datasets to evaluate the effectiveness of SpecAE. The detailed descriptions are shown in Tab. 3.1. Cora is a network consisting of 3,327 scientific publications, with 5,429 links to denote the citation relations. Each article is described by one 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary as node attributes. Pubmed comprises 19, 717 publications with 44,338 citations from biomedical field, with 500 dimension vector descriptions as nodal attributes. We also conduct a case study on another dataset PolBlog [49], which contains various blog content mainly related to political debates written by different liberal bloggers during the Iraq war in 2005.

In order to simulate ground truth outlierness, for Cora and Pubmed, we adopt the same strategy as in [50, 51] to generate a combined set of anomalies from nodal attributes perspective and topological structure perspective. We inject an equal ratio of anomalies for both perspectives. First,

Dataset	Cora	Pubmed
# nodes	3,327	19,717
# edges	5,429	44,338
# attributes	1,433	500
anomaly ratio	5%	5%

Table 3.1: Statistics of Cora and Pubmed.

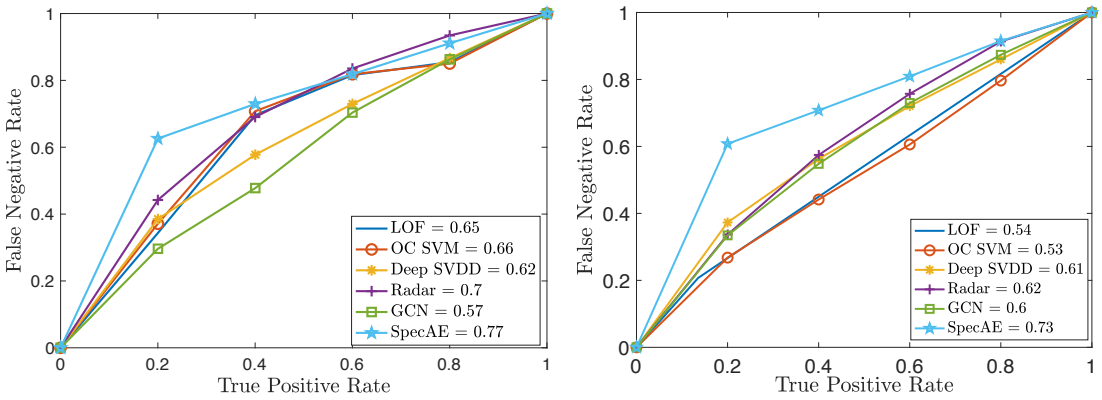


Figure 3.2: ROC and AUC on Cora and Pubmed (from left to right).

we randomly select m bags of words from the word dictionary which have low correlations as new nodal attributes, and replace these generated nodal attributes to the original $\mathbf{X}$, mark them as anomalies; Then we randomly pick another m nodes and replace the attribution of each node i to another node j where the node j deviates from the node i with different context attributes (e.g., in citation networks, node i and j denote different categories of paper), in the meanwhile we keep their original topological relations (we keep the adjacency matrix $\mathbf{A}$ to denote their citation relations).

3.4.2 Baseline Methods

We compare SpecAE with the five baseline methods:

- **LOF** [52]: Detects outliers out of the samples which have a substantially lower density. LOF based methods compute the local density deviation of nodal attributes with respect to their neighbors.

- **OC-SVM** [53]: Estimates the boundary of distribution that encompasses most of the observations (nodal attributes), and then labels the outliers which lie far from the learned boundary.

- **Deep-SVDD** [8]: Trains a neural network while minimizing the volume of a hypersphere that encloses the network representations of the nodal attributes, analyzes the abnormality of each node based on the distance between the nodal attributes and the center of the learned sphere.

- **Radar** [13]: Detects anomalies whose behaviors are different from the majority by characterizing attribute reconstruction residuals and its correlation with topological relations and nodal attributes.

- **GCN** [43]: Combines graph structures and nodal features in the convolution, and propagates over the graph through multiple layers. In our experiment, we modify the structure of the original GCN to accommodate the task of anomaly detection. We add the same density estimation layers as SpecAE after the graph convolutions for fair comparisons.

As shown in the introduction above, we compare SpecAE with three types of baselines. First, to investigate the effectiveness of utilizing features from both topological dependencies and nodal attributes instead of from single modality, we include LOF and OC-SVM. Second, to show the capacity of deep structures, we compare the performance between the shallow models (LOF, OC-SVM and Radar), as well as deep structured methods (Deep-SVDD and GCN). Third, to show the effectiveness of the proposed SpecAE, we modify the vanilla GCN for the specific anomaly detection task.

3.4.3 Experimental Results

We evaluate the anomaly detection performance by comparing SpecAE with the five baseline methods. Accuracy@K is utilized as the evaluation metric and the results are reported in Tab. 3.2. We output a ranking list based on the anomalous degree of different nodes, and we measure the

Accuracy@K								
	Cora				Pubmed			
K	5	10	15	20	5	10	15	20
LOF	85.60	80.91	76.62	72.01	85.71	81.27	76.89	72.71
OC-SVM	85.45	80.76	76.92	72.45	85.54	81.03	76.47	72.27
Deep-SVDD	87.89	85.49	81.50	77.25	87.93	84.63	80.96	77.07
Radar	87.70	85.27	82.35	78.58	86.02	81.90	78.03	74.06
GCN	85.23	83.38	76.70	71.05	86.95	83.57	80.11	76.41
SpecAE	**94.31**	**90.81**	**86.67**	**82.05**	**94.98**	**90.50**	**85.96**	**81.52**

Table 3.2: Accuracy@K performance comparisons on two datasets.

effectiveness of every detection method in its top K ranked nodes. Then, we report the results in terms of ROC-AUC in Fig. 3.2. For both metrics, SpecAE consistently achieves good performance. In addition, we make the following observations. (1) In general, the proposed SpecAE outperforms all of the baseline methods. Comparing with the methods which only utilize uni-modal information (nodal attributes in LOF, OC-SVM and Deep-SVDD), our proposed approach achieves better performance which validates the importance of applying heterogeneous multi-modal sources (nodal attributes and topological relations). (2) The performance of deep models (Deep-SVDD, GCN and SpecAE) are superior to the conventional anomaly detection methods in shallow structures (LOF and OC-SVM). It verifies the effectiveness of extracting features to represent the nodal attributes on attributes networks in deep structures which can extract features in a more effective way. (3) SpecAE shows the importance of enlarging the model capacity by introducing the mutual interactions between nodal attributes with linkage connections and enabling topological relations to contain additional information. As can be observed, Spectral autoencoder can achieve better performance on anomaly detection application than pure GCN based model, which justifies the advantage of our proposed model that overcomes the drawbacks of the GCN, and adopts it into an anomaly detection scenario. We will quantitatively evaluate the specific properties of the SpecAE structure in the ablation analysis part.

Metric	Accuracy	Precision	Recall	F1	AUC
SpecAE-S	54.51	12.54	**59.63**	20.72	58.45
SpecAE-N	90.51	52.40	52.59	52.50	73.65
SpecAE-nr	82.24	11.07	11.11	11.09	50.53
SpecAE $\alpha = 1$	90.36	51.66	51.85	51.76	73.96
SpecAE $\alpha = 0.7$	**91.93**	**54.98**	55.19	**55.08**	**77.15**

Table 3.3: Ablation and hyperparameter analysis.

3.4.4 Ablation Studies

We conduct ablation studies to demonstrate the importance of individual components in SpecAE. We perform ablation analysis comparing SpecAE with four alternatives. (i) SpecAE-S: SpecAE without the representations for community anomaly detection, i.e., $\mathbf{Z}_{\mathcal{G}}$ and $\mathbf{Z}_{\mathcal{G}\text{error}}$; (ii) SpecAE-N: SpecAE without the representations for global anomaly detection, i.e., $\mathbf{Z}_X$ and $\mathbf{Z}_{X\text{error}}$; (iii) SpecAE-nr: SpecAE without the reconstruction components $\mathbf{Z}_{X\text{error}}$ and $\mathbf{Z}_{\mathcal{G}\text{error}}$; and (iv) training with an extreme value hyper-parameter $\alpha = 1$ (short-circuit the self features in the graph convolution and deconvolution). Results on Cora are shown in Tab. 3.3, indicating that the comprehensive method is superior to the variants, since it outperforms most of the ablation test settings where one component has been removed or replaced at a time from the full system and performed re-training and re-testing. Similar results could be observed on other datasets (Pubmed). Due to the reason of the space limit, we omit the detailed analysis. According to the results, we can find that the context information $\mathbf{Z}_{\mathcal{G}}$ provided by the SpecAE is crucial for anomaly detection in the attributed network (SpecAE-N, SpecAE outperform the other structures by 15% in AUC). In the meanwhile, when we eliminate the $\mathbf{Z}_X$, the performance drops by 3.5%. Removing the reconstruction process hurts the performance by 27%, which validates the reconstruction introduces useful information as indicators for anomaly detection.

3.4.5 A Case Study

We conduct a case study using the PolBlog dataset. The attributes of each node represent words appeared in each blog based on a given word dictionary. As shown in the Tab. 3.4, six

(Anomaly).com	Abnormal Keywords
thismodernworld	siberians, fenno, scandinavia, ultraviolet, primates, colder
directorblue.blogspot	boiler, inflatable, choppers, streamline, rooftop, jettison, sheathed
fafblog.blogspot	cheesecake, pies, crusts, barbecues, carrots, grapefruit pizzas
balloon-juice	recapturing, wooly, buffy, talons, snarled, tails, dismemberment
fringeblog	mosquitos, nerf, redwoods, mammoths, fungi, snail, hawked
nomayo.mu.nu	pancakes, cheese, stewed, pork, brunch

Table 3.4: Keywords selected from the anomalous nodes on PolBlog.

sampled anomalies (nodes) are listed for more specific explanations. The abnormal keywords refer to the key information which appears in the anomaly bloggers, yet has low frequency and lacks political flavor in a global perspective among all bloggers. To be more specific, some of the posts actually focus on other fields like climates (thismordernworld), food (nomayo.mu.nu, fafblog.blogspot), constructions and tools (directorblue.blogspot), huntings and creatures (ballon-juice, fringeblog). Comparing with the most popular political topics, these selected keywords in the abnormal bloggers are less relevance to the political events.

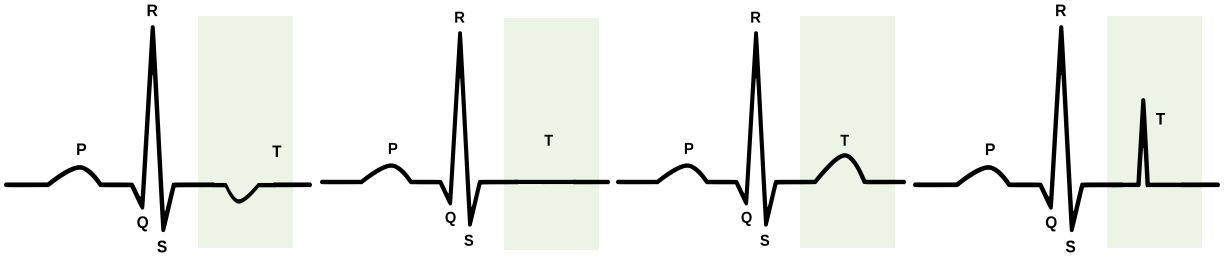
4. DISENTANGLED REPRESENTATION ON TIME SERIES

Time-series representation learning is a fundamental task for time-series analysis. While promising progress has been made toward learning accurate representations for downstream applications, the learned representations of existing methods often lack interpretability, and do not encode semantic meanings, due to their entangled feature spaces. In this chapter, we aim to extract the semantic-rich temporal correlations underlying the latent factorized representations. Inspired by the success of disentangled representation learning in computer vision, we investigate the possibility of learning semantic-rich representations of time-series, which remains unexplored for three main challenges: 1) the sequential data structure introduces complex temporal dependencies and makes the latent representations hard to interpret; 2) sequential models suffer from KL vanishing problem; and 3) interpretable semantic concepts for time-series often rely on multiple factors instead of individuals. To handle them, we propose **D**isentangle **T**ime-**S**eries (`DTS`), a novel disentanglement enhancement framework for sequential data. Specifically, to generate hierarchical semantic concepts as the interpretable and disentangled representation of time-series, `DTS` introduces multi-level disentanglement strategies by covering both individual latent factors and group semantic segments. We further propose an evidence lower bound (ELBO) decomposition approach to balance the preference between correct inference and fitting data distribution problems. We theoretically show how to alleviate the KL vanishing problem: `DTS` introduces a mutual information maximization term, while preserving a heavier penalty on the total correlation and the dimension-wise KL to keep the disentanglement property. Experimental results on various real-world datasets demonstrate that the representations learned by `DTS` achieve superior performance in downstream applications, with high interpretability of semantic concepts.

4.1 Introduction

Unsupervised time-series representation learning, as a fundamental task of time-series analysis, aims to extract low-dimensional representations from complex raw time-series without human

(a) Semantic factors for images, where a semantic factor controls the eye glasses of a human facial image [54].



(b) Semantic factors of time-series, where a semantic factor controls the sequential trend of a time-series.

Figure 4.1: Two traversal plot examples of disentanglement.

supervision. These representations can then be used to benefit many downstream tasks, such as classification, clustering [55], and anomaly detection [56]. Recently, deep generative models have shown great representation ability in modeling complex underlying distributions of time-series data. The most representative examples include LSTM-VAE and its different variants [57, 58, 59].

While these representation learning techniques have achieved good performance in many downstream applications, the learned representations often lack the interpretability to expose tangible semantic meanings. In many cases, especially in high-stake domains, an interpretable representation is critical for diagnosis or decision-making. For example, learning interpretable and semantic-rich representations is useful for decomposing the ECG into cardiac cycles with recognizable phases as independent factors. Furthermore, extracting and analyzing common sequential patterns ($i.e.$, normal sinus rhythms) from massive ECG records may assist clinicians to better understand the irregular symptoms (like sinus tachycardia, sinus bradycardia, and sinus arrhythmia). In contrast,

diagnostic processes without transparency or accurate explanations may lead to suboptimal or even risky treatments.

To enable semantically meaningful representations, researchers in computer vision have been devoted to learning disentangled representations, which decompose the representations into subspaces and encode them as separate dimensions [60]. A disentangled representation can be defined as one where single latent units are sensitive to changes in a single latent factor while being relatively invariant to changes in other factors; that is, different dimensions in the latent space are probabilistically independent. Fig. 4.1 (a) shows an example, where a semantic factor controls the eyeglasses of a human facial image. Learning factors of variations in the images enables the emergence of semantic meanings in the underlying distribution [54]. Motivated by the success of disentanglement in the image domain, in this work, we explore disentangled representations for time-series data. Fig. 4.1 (b) shows an example of how the learned semantic factor can control the shape of ECG time-series. Medically, inverted, biphasic, or flattened T wave could also provide insights into the abnormalities of the ventricular repolarization or secondary to abnormalities in ventricular depolarisation. In addition, the QT interval as a group of individual patterns from the beginning of the Q wave to the end of the T wave, could represent the physiologic reactions for the ventricles of the heart to depolarize and repolarize. Thus, there exists a vital need for the methods that can enhance the interpretability of time-series representations from the perspectives of both single factor disentanglement and group-level factor disentanglement, to induce multi-level semantics.

However, time-series presents great challenges for learning disentangled representations. Firstly, *complex data structure with temporal correlations makes latent codes hard to interpret*. Time-series data usually contain temporal correlations, which cannot be directly captured and interpreted by traditional image-focused disentangle methods [54, 61, 62]. While traditional sequential models, like LSTM or LSTM-VAE, could be used to model the temporal correlations, they neither provide interpretable predictions, as is often criticized, nor have an disentangle mechanism. Secondly, *sequential model may suffer from KL vanishing problem*. Recent work [63] shows that:

since the LSTM generative model often has strong expressiveness, the reconstruction term in the objective will dominate the KL divergence term. In this case, the model would generate time-series without making effective use of the latent codes. Besides, the latent variables Z will become independent of the observations $\mathbf{x}$ when the KL divergence term collapses to zero. It's referred to as the *information preference* problem [64]. Thirdly, *interpretable semantic concepts often rely on multiple factors instead of individuals*. A human-understandable sequential pattern, called semantic components, is usually correlated with multiple factors. It is hard to interpret time-series with a single latent factor and its variations.

To address these challenges, in this chapter, we propose **D**isentangle **T**ime-**S**eries (DTS) for learning semantically interpretable time-series representations. To the best of our knowledge, DTS is the first attempt to incorporate disentanglement strategies for time-series. It is one of the first to interpret time-series representations with state-of-the-art deep models. In particular, we design a multi-level time-series disentanglement strategy that accounts for both individual latent factor and group-level semantic segments, to generate hierarchical semantic concepts as the interpretable and disentangled representations of time-series. For individual latent factor disentanglement, DTS introduces a mutual information maximization term (MIM) to encourage high mutual information between the latent codes and the original time-series. By leveraging the MIM, DTS preserves the disentanglement property via evidence lower bound (ELBO) decomposition, while alleviates the KL vanishing problem. We further theoretically prove that the decomposition process can balance the preference between correct inference and fitting data distribution. For group-level semantic segment disentanglement, DTS learns decomposed semantic segments that contain batches of independent latent variables via applying gradient reversal layers on irrelevant tasks. The learned group segment disentanglement can benefit many downstream tasks such as domain adaptation. Extensive experiments on real-world datasets demonstrate that DTS could provide more meaningful disentangled representations of time-series, and is quantitatively effective for downstream tasks.

The contributions of this work are summarized as follows:

- We introduce a novel and challenging problem ($i.e.$, disentangle time-series) and propose DTS, to incorporate disentanglement strategies for time-series representation learning task.
- We devise multi-level time-series disentanglement strategies, covering both individual latent factor and group-level semantic segments, to generate hierarchical semantic concepts as the interpretable and disentangled representation of time-series.
- We propose an ELBO decomposition strategy to balance the preference between correct inference and fitting data distribution problem. We theoretically show how to alleviate the KL vanishing problem via introducing the mutual information term. Meanwhile, we preserve a heavier penalty on the total correlation and the dimension-wise KL to keep the disentangle property.
- We conduct extensive experiments on the time-series representation and the domain-adaptation tasks, and provide insights on interpreting the latent representations with semantic meanings.

4.2 Methodology

In this section, we propose DTS (see Fig. 4.2), a multi-level disentanglement approach to enhance time-series representation learning. The key idea of DTS is that the latent space should involve multiple independent factors as semantic concepts rather than conjugated representation. We first introduce the disentanglement problem (Section 2.1) and the challenges (Section 2.2). Then, we introduce an individual disentanglement approach for learning interpretable representations (Section 2.3). Finally, we introduce how to learn disentangled group segments as semantic components (Section 2.4).

4.2.1 Problem Statement

To facilitate understanding, we focus on univariate time-series in the definitions; one can easily extend them to multivariate time-series.

Notations: We use lowercase alphabet $x \in \mathbb{R}$ to denote a scalar value in a time-series, $z \in \mathbb{R}$ to denote a scalar latent variable, lowercase boldface letter $\mathbf{x} = [x_1, x_2, ..., x_T] \in \mathbb{R}^T$ to denote a

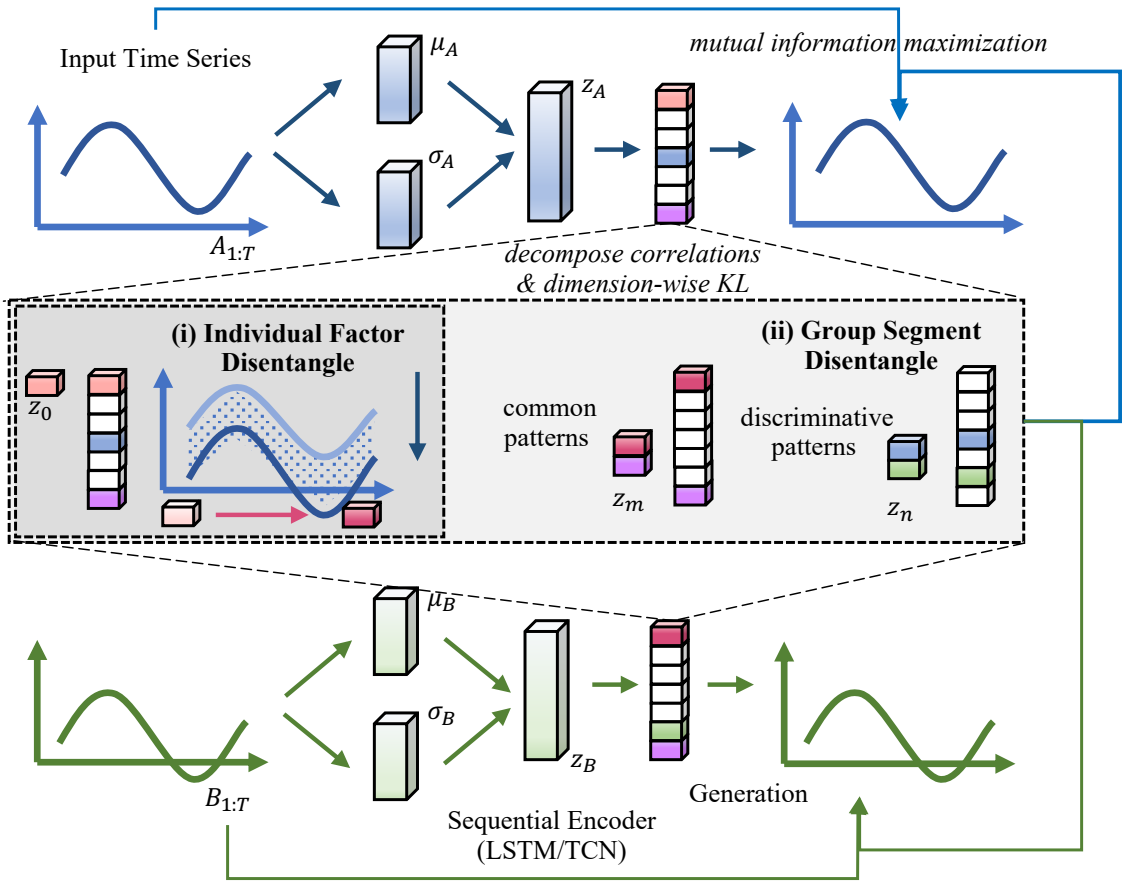


Figure 4.2: An overview of DTS.

time-series of length T, lowercase boldface letter $\mathbf{z} = [z_1, z_2, ..., z_N] \in \mathbb{R}^N$ to denote a segment of latent variables with size N, uppercase alphabet Z to denote a set of latent variables, which either consists of segments, *e.g.*, $Z = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_N\}$, or consists of scalars, *e.g.*, $Z = \{z_1, z_2, ..., z_N\}$, and $|Z|$ to denote the size of all the latent variables, *i.e.*, the dimension of the representations.

Disentanglement Problem: The independent latent factors and semantic segments can be defined as follows. [Independent Latent Factors and Segments] Two latent variables z_i and z_j (or semantic segments $\mathbf{z}_i$ and $\mathbf{z}_j$) are independent if the change of the sequential patterns corresponding to variable z_i (segment $\mathbf{z}_i$) is relatively invariant to variable z_j (segment $\mathbf{z}_j$) and vice versa, denoted as $z_i z_j$ ($\mathbf{z}_i \mathbf{z}_j$).

Based on Definition 4.2.1, we formally define the multi-level time-series disentanglement problem as follows.

- *Individual Latent Factor Disentanglement*: We aim to learn a set of decomposed latent variables $Z = \{z_1, z_2, ..., z_N\}$, such that all the pairs of latent variables in Z are independent, *i.e.*, $z_i \perp z_j, \forall i, j$, where $i \in \{1, 2, ..., N\}$, $j \in \{1, 2, ..., N\}$, and $i \neq j$.

- *Group Segment Disentanglement*: We aim to learn a set of decomposed semantic segments $Z = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_N\}$, such that all the pairs of segments in Z are independent, *i.e.*, $\mathbf{z}_i \perp \mathbf{z}_j, \forall i, j$, where $i \in \{1, 2, ..., N\}$, $j \in \{1, 2, ..., N\}$, and $i \neq j$.

4.2.2 β-VAE and KL Vanishing Problem

In this section, we focus on the latent codes disentanglement task using the generative model family. Generative models have shown superior performance in learning the complex distributions. LSTM-VAE [65] is widely used in time-series analytics. A latent variable generative model defines a joint distribution between a feature space $Z \in \mathcal{Z}$, and the observation space $\mathbf{x}_{1:T} \in \mathcal{X}$. Suppose we have a simple prior distribution $p(Z)$ is placed over the latent variables. The distribution of the observations conditioned on latent variables is modeled with a deep network $p_\theta(\mathbf{x}_{1:T} \mid Z)$. If we have the true underlying distribution as $p_\mathcal{D}(\mathbf{x})$ (empirically approximated through the training set), then a natural training objective is to maximize the marginal likelihood as:

$$\mathbb{E}_{p_\mathcal{D}(\mathbf{x})} \left[\log p_\theta(\mathbf{x})\right] = \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T})} \left[\log \mathbb{E}_{p(Z)} \left[p_\theta(\mathbf{x}_{1:T} \mid Z)\right]\right]. \tag{4.1}$$

Directly optimizing the likelihood is generally intractable, since computing $p_\theta(\mathbf{x}) = \int_z p_\theta(\mathbf{x} \mid Z)p(Z)\mathrm{d}z$ requires integration. A widely-used solution [66] is to approximate the posterior via an amortized inference distribution $q_\phi(Z \mid \mathbf{x})$ and jointly optimize a lower bound to the log likelihood as:

$$\begin{aligned} \mathcal{L}_{\mathrm{ELBO}}(\mathbf{x}) &= -D_{\mathrm{KL}} \left(q_\phi(Z|\mathbf{x}_{1:T}) \| p(Z)\right) + \mathbb{E}_{q_\phi(Z|\mathbf{x}_{1:T})} \left[\log p_\theta(\mathbf{x}_{1:T}|Z)\right] \\ &\leq \log p_\theta(\mathbf{x}_{1:T}). \end{aligned} \tag{4.2}$$

A disentangled representation [67] can be defined as one where single latent units are sensitive to changes in single generative factor, while being relatively invariant to changes in other factors. β-VAE [67] is a variant of the variational autoencoder that attempts to learn a disentangled repre-

sentation by optimizing a heavily penalized objective with $\beta > 1$. The penalization is capable of obtaining models exhibiting disentangled effects in image datasets. With increasing β, the latent codes could be more disentangled by allowing distributions in the latent space to be further away from each other by less satisfying the KL-term constraint to fit the marginally Gaussian distribution. The semantically similar observations might be mapped closer, thus creating a sort of clusters for interpretations.

$$\begin{aligned}\mathcal{L}_{\beta-\mathrm{ELBO}}(\mathbf{x}) &= -\beta D_{\mathrm{KL}}\left(q_\phi(Z|\mathbf{x}_{1:T})\|p(Z)\right) + \mathbb{E}_{q_\phi(Z|\mathbf{x}_{1:T})}\left[\log p_\theta(\mathbf{x}_{1:T}|Z)\right] \\ &\leq \log p_\theta(\mathbf{x}_{1:T}).\end{aligned} \tag{4.3}$$

However, for sequential time-series data, prior work [63] shows that: the LSTM generative model often has a strong expressiveness. The reconstruction term in the objective will dominate the KL divergence term during the training phase. In this case, the model will generate time-series without effectively making use of latent codes. Besides, the latent variable Z becomes independent from observations $\mathbf{x}$, when the KL divergence term collapses to zero. As a result, the latent variable Z can not serve as an effective representation for the input $\mathbf{x}$, and it can not be used for downstream tasks such as classification and clustering. It can be further referred to as the *information preference* problem [64]. Thus, pushing Gaussian clouds away from each other in the latent space becomes meaningless if latent distributions are unhooked with the observation space.

4.2.3 Individual Latent Factor Disentanglement

To alleviate the KL vanishing problem, as well as to preserve the disentangle property, in this section, we introduce the mutual information maximization term to the ELBO decomposition. The goal is to learn a better representation Z that could more effectively capture the semantic characteristics of the input $\mathbf{x}$.

4.2.3.1 ELBO TC-Decomposition

To further understand the internal mechanism within the disentangle strategy, we decompose the evidence lower bound (ELBO) to find evidence linking factorial representations to disentanglement.

Mathematically, the KL term in Eq. (4.2) and 4.3 can be decomposed with a factorized $p(Z)$ as:

$$\begin{aligned} D_{\mathrm{KL}}(q(Z \mid \mathbf{x}_{1:T}) || p(Z)) = & \underbrace{\mathrm{KL}(q(Z, \mathbf{x}_{1:T}) || q(Z) p(\mathbf{x}_{1:T}))}_{\text{(i) Index-Code MI}} \\ & \underbrace{+ \mathrm{KL}(q(Z) \| \prod_j q\left(z_j\right))}_{\text{(ii) Total Correlation}} \underbrace{+ \sum_j \mathrm{KL}\left(q\left(z_j\right) \| p\left(z_j\right)\right)}_{\text{(iii) Dimension-wise KL}} \end{aligned} \tag{4.4}$$

where z_j denotes the jth dimension of the latent variable.

To illustrate how to get the decomposition of the KL term in Eq. (4.2) and 4.3, the KL term can be further decomposed with a factorized $p(Z)$ step-by-step as:

$$\begin{aligned} & \frac{1}{N} \sum_{n=1}^{N} \mathrm{KL}\left(q\left(Z \mid \mathbf{x}_{1:T}^n\right) \| p(Z)\right) = \mathbb{E}_{p(\mathbf{x}_{1:T}^n)}[D_{\mathrm{KL}}(q(Z \mid \mathbf{x}_{1:T}^n) \| p(Z))] \\ = & \quad \mathbb{E}_{q(Z, \mathbf{x}_{1:T}^n)}\left[\log \frac{q(Z \mid \mathbf{x}_{1:T}^n)}{q(Z)}\right] + \mathbb{E}_{q(Z)}\left[\log \frac{q(Z)}{\prod_j q\left(z_j\right)}\right] \\ & \quad + \mathbb{E}_{q(Z)}\left[\sum_j \log \frac{q\left(z_j\right)}{p\left(z_j\right)}\right] \\ = & \quad \mathbb{E}_{q(Z, \mathbf{x}_{1:T}^n)}\left[\log \frac{q(Z \mid \mathbf{x}_{1:T}^n) p(\mathbf{x}_{1:T}^n)}{q(Z) p(\mathbf{x}_{1:T}^n)}\right] + \mathbb{E}_{q(Z)}\left[\log \frac{q(Z)}{\prod_j q\left(z_j\right)}\right] \\ & \quad + \sum_j \mathbb{E}_{q(Z)}\left[\log \frac{q\left(z_j\right)}{p\left(z_j\right)}\right] \\ = & \quad \mathbb{E}_{q(Z, \mathbf{x}_{1:T}^n)}\left[\log \frac{q(Z \mid \mathbf{x}_{1:T}^n) p(\mathbf{x}_{1:T}^n)}{q(Z) p(\mathbf{x}_{1:T}^n)}\right] + \mathbb{E}_{q(Z)}\left[\log \frac{q(Z)}{\prod_j q\left(z_j\right)}\right] \\ & \quad + \sum_j \mathbb{E}_{q(z_j) q\left(Z_{\setminus j} \mid z_j\right)}\left[\log \frac{q\left(z_j\right)}{p\left(z_j\right)}\right] \\ = & \quad \mathbb{E}_{q(Z, \mathbf{x}_{1:T}^n)}\left[\log \frac{q(Z \mid \mathbf{x}_{1:T}^n) p(\mathbf{x}_{1:T}^n)}{q(Z) p(\mathbf{x}_{1:T}^n)}\right] + \mathbb{E}_{q(Z)}\left[\log \frac{q(Z)}{\prod_j q\left(z_j\right)}\right] \\ & \quad + \sum_j \mathbb{E}_{q(z_j)}\left[\log \frac{q\left(z_j\right)}{p\left(z_j\right)}\right] \\ = & \quad D_{\mathrm{KL}}(q(Z, x_{1:T}) \| q(Z) p(\mathbf{x}_{1:T})) + D_{\mathrm{KL}}(q(Z) \| \prod_j q\left(z_j\right)) \\ & \quad + \sum_j D_{\mathrm{KL}}\left(q\left(z_j\right) \| p\left(z_j\right)\right) \end{aligned} \tag{4.5}$$

where z_j denotes the jth dimension of the latent variable.

The first term can be interpreted as the index-code mutual information (MI). The index-code MI is the mutual information $I_{q_\phi}(Z; \mathbf{x})$ between the data variable and latent variable based on the empirical data distribution $q_\phi(Z, \mathbf{x})$. From the information theory, the second term is referred to as the total correlation (TC). TC acts as one of many generalizations of mutual information to more than two random variables [68]. TC is also to measure the dependencies between the variables. The penalty on TC forces the model to find statistically independent factors in the data distribution. A heavier penalty on this term induces a more disentangled representation, and that the existence of this term is the reason β -VAE has been successful. One recent study empirically verifies [62, 69] that TC is the most important term in this decomposition for learning disentangled representations by only penalizing on this term. The last term is referred to as the dimension-wise KL. This term mainly prevents individual latent dimensions from deviating too far from their corresponding priors. It plays a role as a complexity penalty on the aggregate posterior, which follows from the minimum description length [70] formulation of the ELBO.

By decomposing the ELBO into separate components, we can have a new perspective for the problem when the latent codes are becoming independent from observations. Introducing a heavier penalty on the ELBO tends to neglect the mutual information between Z and $\mathbf{x}$. Thus, the mutual information becomes vanishingly small. Intuitively, the reason is that the learned $p_\theta(\mathbf{x}|Z)$ is the same for all $z \in \mathcal{Z}$, implying that the Z is not dependent on the observations $\mathbf{x}$. This is undesirable because a major goal of unsupervised learning is to learn meaningful latent features that should depend on the observations. If we directly apply β-VAE models to time-series data for disentanglement, β-VAE can not effectively trade-off weighting of the mutual information and the reconstruction. Desired disentangled representations emerge when the right balance is found between information preservation (reconstruction cost as regularisation) and latent channel capacity restriction (when $\beta > 1$). Increasing the β may intensify the mutual information vanishing problem: when the model has a better quality of disentanglement within the learned latent representations, it penalizes the mutual information simultaneously. This, in turn, can lead to under-fitting or

ignoring the latent variables.

4.2.3.2 ELBO DTS-Decomposition

We now introduce how to balance the preference between correct inference and fitting data distribution, and provide a new ELBO decomposition solution that sheds light on solving the information preference problem.

Previously, we discussed that optimizing the ELBO tends to push the probability mass of $q_\phi(Z \mid \mathbf{x})$ too far from 0. This tendency is a property of the ELBO objective and true for any x and Z. However, this is made worse by the fact that x is often higher dimensional compared to Z, so any error in fitting x will be magnified compared to Z. To encourage the model to use the latent codes, we add a mutual information maximization term, which encourages a high mutual information between $\mathbf{x}$ and Z. In other words, we can address the information preference problem through balancing the preference between correct inference and fitting data. Comparing with the ELBO in LSTM-VAE (in Eq. (4.2)), we can rewrite it as:

$$\begin{aligned}\mathcal{L}_{\mathrm{ELBO}}(\mathbf{x}) = &-D_{\mathrm{KL}}\left(q_\phi(Z \mid \mathbf{x}_{1:T}) \| p(Z)\right) + \alpha I_{q_\phi}(\mathbf{x}_{1:T}; Z) \\ &+ \mathbb{E}_{q_\phi(Z|\mathbf{x}_{1:T})}\left[\log p_\theta(\mathbf{x}_{1:T} \mid Z)\right]\end{aligned} \tag{4.6}$$

where $I_{q_\phi}(\mathbf{x}; Z)$ denotes the mutual information between $\mathbf{x}$ and Z under the distribution $q_\phi(\mathbf{x}; Z)$. But the objective can not be directly optimized. Thus, we rewrite it into another equivalent form:

$$\begin{aligned}\mathcal{L}_{\mathrm{ELBO}}(\mathbf{x}) = &-D_{\mathrm{KL}}\left(q_\phi(Z \mid \mathbf{x}_{1:T}) \| p(Z)\right) + \alpha D_{\mathrm{KL}}\left(q_\phi(Z) \| p(Z)\right) \\ &+ \mathbb{E}_{q_\phi(Z|\mathbf{x}_{1:T})}\left[\log p_\theta(\mathbf{x}_{1:T} \mid Z)\right].\end{aligned} \tag{4.7}$$

We find that the mutual information maximization term (the second part of Eq. 4.7) plays the same role as the first term in the ELBO-TC decomposition (as shown in Eq. (4.4)). But the optimization directions are contrary. Thus, increasing the disentanglement degree may intensify the KL vanishing problem, and vice versa. To enforce the model to preserve the disentangle property while alleviating the KL vanishing, here, we combine the mutual information regularizer term with the ELBO-TC decomposition in Eq. (4.4) and merge the mutual information maximization term,

then the ELBO can be written as:

$$\begin{aligned}\mathcal{L}_{\mathrm{ELBO}}(\mathbf{x}) = &-\beta D_{\mathrm{KL}}(q(Z)\|\prod_j q\left(z_j\right)) - \beta\sum_j D_{\mathrm{KL}}\left(q\left(z_j\right)\|p\left(z_j\right)\right) \\ &+ (\alpha-\beta)D_{\mathrm{KL}}\left(q_\phi(Z)\|p(Z)\right) + \mathbb{E}_{q_\phi(Z|\mathbf{x}_{1:T})}\left[\log p_\theta(\mathbf{x}_{1:T} \mid Z)\right],\end{aligned} \tag{4.8}$$

where the mutual information regularizer merges with the first term in the decomposition analysis. Mathematically, we alleviate the KL vanishing problem by introducing the mutual information maximization term, while preserve a heavier penalty (when $\beta > 1$) on the total correlation and the dimension-wise KL to keep the disentangle property.

4.2.4 Latent Group Segment Disentanglement

The individual factor disentanglement introduced in the last section can't guarantee interpretability. For time-series applications, it's hard to interpret the single latent factor, and human-interpretable semantic concepts typically rely on multiple latent factors instead of one single factor. Thus, in this section, we introduce how to disentangle latent codes of time-series into group segments (Section 4.2.4.1). We further show that our proposed group segment disentanglement could benefit the domain adaptation task (Section 4.2.4.2).

4.2.4.1 Group Segment Disentanglement

Different from individual variable analysis, in this section, we aim to learn decomposed semantic segments that contain batches of latent variables. One main advantage of the group segment disentanglement is to extract more than one latent factors, which simultaneously contribute to one complex sequential semantic concept.

Assume that the semantic segments are independent factors, *i.e.*, $\mathbf{z}_m \perp \mathbf{z}_n$ with every pairs of m, n sampled from $\mathbf{z} = \{\mathbf{z}_1 ... \mathbf{z}_k\}$, where k is the number of the segments. For ease of presentation, we take m, n as two semantic segments for illustration. Our method could be easily extended to multiple segments scenarios. Given $\mathbf{x}$, the new time-series is generated from two independent latent group segments, *i.e.*, $\mathbf{z}_m$ encodes the m_{th}-segment and $\mathbf{z}_n$ encodes the n_{th}-segment.

Theorem 4. *Assume that two group segments are independent, i.e.,* $\mathbf{z}_m \perp \mathbf{z}_n$*, and let* $\mathbf{z} = \{\mathbf{z}_m, \mathbf{z}_n\}$*,*

the empirical error on the disentangled segments with a hypothesis h is:

$$\epsilon(h) = \mathrm{E}_{\mathbf{z}_m \sim \mathcal{Z}} \left[C_m \left(\mathbf{z}_m \right) - h \left(\mathbf{z}_m \right) \right] + \mathrm{E}_{\mathbf{z}_n \sim \mathcal{Z}} \left[C_n \left(\mathbf{z}_n \right) - h \left(\mathbf{z}_n \right) \right] \tag{4.9}$$

where $\epsilon_{\mathrm{T}}^{\mathrm{y}}(h)$ denotes the empirical error of `DTS` *with respect to h.*

Proof. Since $\mathbf{z}_m \perp \mathbf{z}_n$, we can derive the empirical error as follows:

$$\begin{aligned} \epsilon(h) &= \mathrm{E}_{(\mathbf{z}_m, \mathbf{z}_n) \sim \mathcal{Z}} \left[C \left(\mathbf{z} \right) - h \left(\mathbf{z} \right) \right] \\ &= \mathrm{E}_{\mathbf{z}_m \sim \mathcal{Z}} \left[C_m \left(\mathbf{z}_m \right) - h \left(\mathbf{z}_m \right) \right] + \mathrm{E}_{\mathbf{z}_n \sim \mathcal{Z}} \left[C_n \left(\mathbf{z}_n \right) - h \left(\mathbf{z}_n \right) \right]. \end{aligned} \tag{4.10}$$

Based on the independence property between $\mathbf{z}_m$ and $\mathbf{z}_n$, the distribution of $\mathcal{Z}$ can be decomposed into two parts so as to the error. Following the evidence lower bound of the marginal likelihood in the Eq. (4.7), we get a similar form for group segments:

$$\begin{aligned} \mathcal{L}_{\mathrm{ELBO-G}}(\mathbf{x}) = & -D_{\mathrm{KL}} \left(q_{\phi_m}(\mathbf{z}_m \mid \mathbf{x}_{1:T}) \| p(\mathbf{z}_n) \right) - D_{\mathrm{KL}} \left(q_{\phi_n}(\mathbf{z}_n \mid \mathbf{x}_{1:T}) \| p(\mathbf{z}_n) \right) \\ & + \mathbb{E}_{q_{\phi_m}(\mathbf{z}_m, \mathbf{z}_n \mid \mathbf{x}_{1:T})} \left[\log p_\theta(\mathbf{x}_{1:T} \mid \mathbf{z}_m, \mathbf{z}_n) \right] + \alpha D_{\mathrm{KL}} \left(q_\phi(Z) \| p(Z) \right). \end{aligned} \tag{4.11}$$

We assume that $P\left(\boldsymbol{z}_m\right), P\left(\boldsymbol{z}_n\right) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and ϕ_m and ϕ_n are the parameters of the encoder. By applying a reparameterization trick, we use sequential models (LSTMs or TCNs [71]) $H_m \left(G(\mathbf{x}); \phi_y \right)$ and $H_n \left(G(\mathbf{x}); \phi_d \right)$ as the universal approximator of q to encode the data into $\mathbf{z}_m$ and $\mathbf{z}_n$, respectively.

Meanwhile, we want to make the $\mathbf{z}_m$ task-n-invariant, and make the $\mathbf{z}_n$ task-m invariant. That is, we want to make the $C_m \left(\mathbf{z}_n; \theta_m \right)$ and $C_n \left(\mathbf{z}_m; \theta_n \right)$ less discriminative. To obtain task-invariant representations, we seek the parameters θ_m of the feature mapping that maximize the loss of the q_{θ_m} (by making the distributions of different $\mathbf{z}_m$ as close as possible), while simultaneously seeking the parameters θ_n that minimize the loss of $\mathbf{z}_m$. We aim to solve:

$$\mathbb{E}_{\mathrm{m}} \left(\phi_y, \theta_m, \theta_n \right) = \mathbb{E} \left(C_m \left(\mathbf{z}_m; \theta_m \right), y_m \right) - \lambda \mathbb{E} \left(C_n \left(\mathbf{z}_m; \theta_n \right), y_n \right). \tag{4.12}$$

Moreover, we seek to minimize the loss of $\mathbf{z}_n$. Similarly, we have:

$$\mathbb{E}_{\mathrm{n}}\left(\phi_y, \theta_m, \theta_n\right) = \mathbb{E}\left(C_n\left(\mathbf{z}_n; \theta_n\right), y_n\right) - \lambda \mathbb{E}\left(C_m\left(\mathbf{z}_n; \theta_m\right), y_m\right). \tag{4.13}$$

From Eq. (4.12) and 4.13, we are seeking the parameters $\hat{\theta}_f, \hat{\theta}_y, \hat{\theta}_d$ as:

$$\begin{aligned} \left(\hat{\theta}_f, \hat{\theta}_y\right) &= \arg\min_{\theta_f, \theta_y} E\left(\theta_f, \theta_y, \hat{\theta}_d\right) \\ \hat{\theta}_d &= \arg\max_{\theta_d} E\left(\hat{\theta}_f, \hat{\theta}_y, \theta_d\right), \end{aligned} \tag{4.14}$$

where the parameter λ controls the trade-off between the two objectives that shape the features during training. The update process is very similar to vanilla stochastic gradient descent (SGD) updates for feed-forward deep models. One thing that needs to point out is the $-\lambda$ factor, which tries to make disentangled features less discriminative for the irrelevant task. Here, we use a gradient reversal layer (GRL) [72] to exclude the discriminative information. During the forward propagation, GRL acts as an identity transform. During the backpropagation, GRL takes the gradient from the subsequent level, and multiplies the gradient by a negative constant, then passes it to the preceding layer.

4.2.4.2 Domain Adaptation as Concrete Example

To further illustrate the benefits of the proposed group segments disentanglement for time-series, we apply it to the domain adaptation problem as a concrete application scenario.

In the unsupervised domain adaptation problem, we use the labeled samples $D_S = \left\{\boldsymbol{x}_{1:T}^S, y_i^S\right\}_{i=1}^{n_S}$ on the source domain to classify the unlabeled samples $D_T = \left\{\boldsymbol{x}_j^T\right\}_{j=1}^{n_T}$ on the target domain. We aim to obtain two independent latent variables with disentanglement, including a domain-dependent latent variable $\mathbf{z}_d$ and a class-dependent latent variable $\mathbf{z}_y$. These two variables are expected to encode the domain information and the class information, respectively. Then, we can use the class-dependent latent variable for classification since it is domain-invariant. Mathematically, deriving from Theorem 4, we have:

Theorem 5. *Assume that the class and the domain factors are independent, i.e.,* $\mathbf{z}_y \mathbf{z}_d$. *Let* $\mathbf{z} =$

$\{\mathbf{z}_y, \mathbf{z}_d\}$, *and the error on the disentangled source and target domain with a hypothesis* h *is:*
$\epsilon_S(h) = \mathrm{E}_{\mathbf{z}_y \sim \mathcal{Z}_S}\left[C\left(\mathbf{z}_y\right) - h\left(\mathbf{z}_y\right)\right] + \mathrm{E}_{\mathbf{z}_d \sim \mathcal{Z}_S}\left[C\left(\mathbf{z}_d\right) - h\left(\mathbf{z}_d\right)\right]$

$\epsilon_T(h) = \mathrm{E}_{\mathbf{z}_y \sim \mathcal{Z}_T}\left[C\left(\mathbf{z}_y\right) - h\left(\mathbf{z}_y\right)\right] + \mathrm{E}_{\mathbf{z}_d \sim \mathcal{Z}_T}\left[C\left(\mathbf{z}_d\right) - h\left(\mathbf{z}_d\right)\right].$

According to the Theorem 5, we can find that, the disentangled empirical classification error rate with respect to h in the source domain is lower than before disentanglement ($\epsilon_S^y(h) = \epsilon_S(h) - \epsilon_S^d(h)$, where $\epsilon_S^d(h) \geq 0$). Prior work [73] theoretically proves that the disentanglement of the representation space could be helpful and necessary for obtaining a lower classification error rate. Furthermore, a lower classification error rate on the source domain will tighten the error bound at the target domain.

We measure the discrepancy distance between the source and target distribution with respect to hypothesis h. Formally, we have:

$$d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) = 2 \sup_{h_1, h_2 \in \mathcal{H}} \left| P_{\mathbf{f}\sim\mathcal{S}}\left[h_1(\mathbf{f}) \neq h_2(\mathbf{f})\right] - P_{\mathbf{f}\sim\mathcal{T}}\left[h_1(\mathbf{f}) \neq h_2(\mathbf{f})\right] \right| \tag{4.15}$$

A probabilistic bound on the performance $\varepsilon_{\mathcal{T}}(h)$ of classifier h from $\mathcal{T}$ is evaluated on the target domain, given its performance $\varepsilon_{\mathcal{S}}(h)$ on the source domain, where $\mathcal{S}$ and $\mathcal{T}$ are source and target distributions, respectively:

$$\varepsilon_{\mathcal{T}}(h) \leq \varepsilon_{\mathcal{S}}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) + C \tag{4.16}$$

where C does not depend on a particular h. Prior work [72] proves that optimal discriminator gives the upper bound for the $d_{\mathcal{H}\triangle\mathcal{H}}(\mathcal{S}, \mathcal{T})$, so that tighter bound $\varepsilon_{\mathcal{S}}(h)$ from the source domain could lead to a better approximation of $\varepsilon_{\mathcal{T}}(h)$.

4.3 Related work and Theoretical Justification

In this section, we introduce the related work as well as theoretical justifications of `DTS` comparing to unsupervised disentanglement learning.

4.3.1 Relationship to Existing Models

Here, we discuss the relationship of DTS to existing models based on the information bottleneck principle [74]. The proposed DTS, including individual latent factors and group segments, can be considered as a product of variational decomposition of mutual information terms in the information bottleneck (IB) framework.

4.3.1.1 Information Bottleneck for Unsupervised Models

The unsupervised IB can be treated as a compression process from $\mathbf{x}$ to $\mathbf{z}$ via the parametrized mapping $q_\phi(\mathbf{z} \mid \mathbf{x})$. This process leads to a bottleneck representation $\mathbf{z}$ yet preserving a certain level of information I_x in $\mathbf{z}$ about $\mathbf{x}$. Accordingly, this problem can be formulated as:

$$\min_{\phi: I(\mathbf{Z};\mathbf{X}) \geq I_x} I_{\boldsymbol{\phi}}(\mathbf{X}; \mathbf{Z}), \tag{4.17}$$

and in the Lagrangian formulation as a minimization of:

$$\mathcal{L}(\boldsymbol{\phi}) = I_{\boldsymbol{\phi}}(\mathbf{X}; \mathbf{Z}) - \beta I(\mathbf{Z}; \mathbf{X}). \tag{4.18}$$

4.3.1.2 Relation to LSTM-VAE and β-VAE

Encoders in LSTM-VAE maps a data point from the observation space into a probabilistic output of Gaussian cloud with mean $\mu(\mathbf{x})$ and 'ellipsoid' orientation determined by the diagonal covariance matrix $\mathrm{diag}(\sigma(\mathbf{x}))$. Comparing with LSTM-VAE, β-VAE relaxes the stochastic 'compression' via mapping everything to a Gaussian heap by applying the relaxation parameter that might give more preference to the reconstruction loss. Our proposed DTS can be considered as another form of compression by the minimization of $I_{\boldsymbol{\phi}}(\mathbf{X}; \mathbf{Z})$. DTS relaxes the condition to map all conditional distributions to one Gaussian heap.

4.3.1.3 Relation to Domain Adaptation Methods

Existing domain adaptation methods focus on (i) utilizing maximum mean discrepancy to measure the domain alignment [75]; and (ii) extracting the domain-invariant representation as transferable common knowledge on the feature space [73, 76]. Motivated by the success of disentanglement in the image domain, `DTS` aims to extract the group segments in the latent disentangled semantic representation of the data. `DTS` also introduces interpretability in the latent space via weak-supervised signals as imposing special constraints on the latent codes.

4.3.1.4 Complexity Analysis

`DTS` can disentangle latent factors for sequential data, scale to complicated datasets, and typically requires no more training time than vanilla VAEs. It does not require an exponentially growing computational cost in the number of factors.

4.3.2 Theoretical Possibility

One recent work essentially shows that without inductive biases on both models and datasets, the unsupervised disentangle task is fundamentally impossible [77].

Theorem 6. *Suppose $p(\mathbf{z})$ is a d-dimensional distribution and $d > 1$, let $\mathbf{z} \sim P$ denote any distribution which admits a density $p(\mathbf{z}) = \prod_{i=1}^{d} p(\mathbf{z}_i)$. Then, there exists an infinite family of bijective functions $f : \mathrm{supp}(\mathbf{z}) \to \mathrm{supp}(\mathbf{z})$ such that $\frac{\partial f_i(\boldsymbol{u})}{\partial u_i} \neq 0$ almost everywhere for all i and j (i.e., $\mathbf{z}$ and $f(\mathbf{z})$ are completely entangled) and $P(\mathbf{z} \leq \boldsymbol{u}) = P(f(\mathbf{z}) \leq \boldsymbol{u})$ for all $\boldsymbol{u} \in \mathrm{supp}(\mathbf{z})$ (i.e., they have the same marginal distribution). [77]*

Intuitively, after observing $\mathbf{x}$, we can construct infinitely many generative models, which have the same marginal distribution of $\mathbf{x}$. Any one of these models could be the true causal generative model for the data, and the right model cannot be identified given only the distribution of $\mathbf{x}$.

While Theorem 6 shows that unsupervised disentanglement learning is fundamentally impossible for arbitrary generative models, it is still theoretically possible for our proposed `DTS`. First, from the model structure perspective, our proposed `DTS` exploits TCN and LSTM-like models,

which impose the inductive biases towards preserving contextual information over long sequences in the design of the neural network architecture. Second, from the data structure perspective, time-series as sequential data, contain trend and seasonal patterns, with the serial correlation between subsequent observations. Third, from the implicit supervision perspective, DTS could incorporate additional weak supervision (like signals from the source domain) to guide a better disentangled representation. Thus, we make explicit and implicit inductive bias available during the training phase.

4.4 Experiments

In this section, we conduct experiments to answer research questions as follows:

- **Q1**: Compared with non-disentangled methods, how quantitatively effective is the proposed disentangled strategy?
- **Q2**: Does individual latent factor disentanglement benefit the generation process of the time-series in a more informative way?
- **Q3**: Whether or not latent group segment disentanglement strategy could separate semantic concepts?

4.4.1 Experiment Setup

We provide insights on interpreting the latent representations with semantic meanings: First, to validate the effectiveness of the disentanglement strategy, we conduct DTS on time-series domain adaptation tasks, and further illustrate the benefits of DTS on separating and extracting different semantic meaningful sequential patterns as transferable common knowledge and domain-dependent information (Section 4.2); Second, we provide latent traversals to validate that DTS tends to discover more informative latent factors and provide more meaningful disentangled representations of time series (Section 4.3); Finally, we visualize the disentangled segments over the representation space, to investigate the discriminative ability of the group disentangle strategy.

4.4.1.1 Datasets

We evaluate `DTS` on five benchmark datasets for individual latent factor disentanglement and group segment disentanglement with domain adaptation as a concrete task.

- **Human Activity Recognition (HAR)** [78]: contains sequential accelerometer, gyroscope, and estimated body acceleration data from 30 participants at 50 Hz.
- **Heterogeneity Human Activity Recognition (HHAR)** [79]: includes accelerometer data from 31 smartphones of different manufacturers and models positioned in various orientations.
- **WISDM Activity Recognition (WISDM AR)** [80]: contains 33 participants accelerometer data, which are sampled at 20 Hz.
- **uWave** [81]: is a large gesture library with over 4000 samples collected from eight users over an elongated period of time for a gesture vocabulary with eight gesture patterns.
- **ECG Signal** [82]: contains heartbeats annotated by at least two cardiologists. The annotations are mapped into 5 groups in accordance with the AAMI standard.

Our adaptation problems consist of the realistic use-case adapting a model from one participant's data to another participant's data. Each dataset consists of data from a number of participants. We follow the same procedure in [76] to select 7 of the possible adaptation problems between two domains as [source_id$\rightarrow$ target_id] (excluding adapting a domain to itself).

4.4.1.2 Data Splitting

For the time-series representation task, it is label-free. For the domain adaptation task, the training-test split is 80% and 20% respectively, and the training data is further split into training-validation with the same proportions. We segment the series into non-overlapping windows of 128 time steps. The datasets were stratified by the labels to maintain the same label proportions for training, validation, and testing sets. During the training phase, only the training dataset is

accessible: labeled data for the sources and unlabeled data for the target. During the evaluation, the test data becomes available for quantitative evaluations and visualizations. Before training, all data is normalized to have zero mean and unit variance based on statistics computed from just the training set.

4.4.1.3 Baselines

We compare `DTS` with three state-of-the-art domain adaptation algorithms and one time-series generative model:

- **Recurrent Domain Adversarial Neural Network(R-DANN)** [83]: employs an LSTM network, and promotes the emergence of features that are (i) discriminative for the learning task on the source domain and (ii) indiscriminate with respect to the shift between the domains.
- **Variational Recurrent Adversarial Deep Domain Adaptation (VRADA)** [84]: uses a variational RNN and trains adversarially to capture temporal relationships that are domain-invariant.
- **Convolutional Deep Domain Adaptation (CoDATS)** [76]: leverages domain-invariant domain adaptation methods to operate on time-series data, and utilizes weak supervisions from labels.
- **LSTM-VAE** [65]: is similar to an auto-encoder. It learns an LSTM as the encoder that maps the sequential data to a latent representation in a probabilistic manner, and decodes the latents back to data.

4.4.1.4 Hyperparameter Settings

We summarize the hyperparameters of network architecture, optimizer, and how we train `DTS` and all the baselines as follows.

For individual latent factor disentanglement experiments, we train the `DTS` on the ECG dataset. The framework is shown in Fig. 4.3.

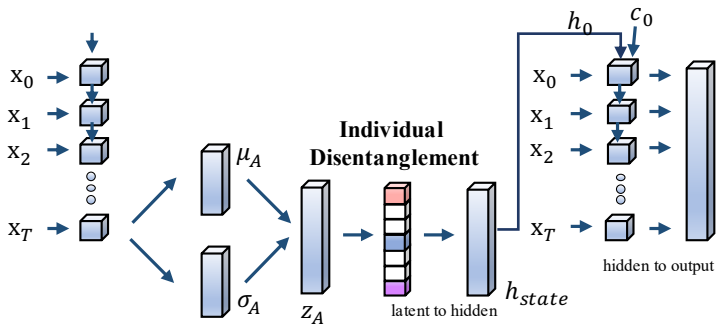


Figure 4.3: The architecture of Individual Latent Factor Disentanglement experiment.

- **LSTM-VAE:** The hidden-size is set to be 90, the depth of the hidden layer is 1, the length of latent variables is set as 12, the batch size is set to be 32, the dropout rate is set to be 0.2. Weights are initialized uniformly in $[-0.1, 0.1]$.

- **Disentanglement:** The disentangle penalty term β is set to be 3, and the mutual information maximization term α is set to be 10.

- **Optimizer:** We train the model by utilizing a batch size of 64 and a momentum of 0.9 with the ADAM optimizer. The learning rate starts at $5e-4$, and is dropped by a factor of 10 at 50% and 75% of the training progress, respectively. The number of epochs has set to be 200, The exactly same optimizer is also adopted in the baselines.

For latent group segment disentanglement experiments, we train the `DTS` on the HAR, HHAR, WISDM AR, and uWave datasets. The framework is shown in Fig. 4.4.

- **TCN:** The depth of the hidden layer is 3, each layer contains a convolution operation as [filters=128, kernel=8], [filters=256, kernel=5], [filters=128, kernel=3], respectively. The activation functions are set to ReLU. The length of latent variables is set to be 12, the batch size is set to be 32, the dropout rate is set to be 0.2. The dropout rate is set to be 0.1. Weights are initialized uniformly in $[-0.1, 0.1]$.

- **Disentanglement:** The disentangle penalty term β is set to be 3, and the mutual information

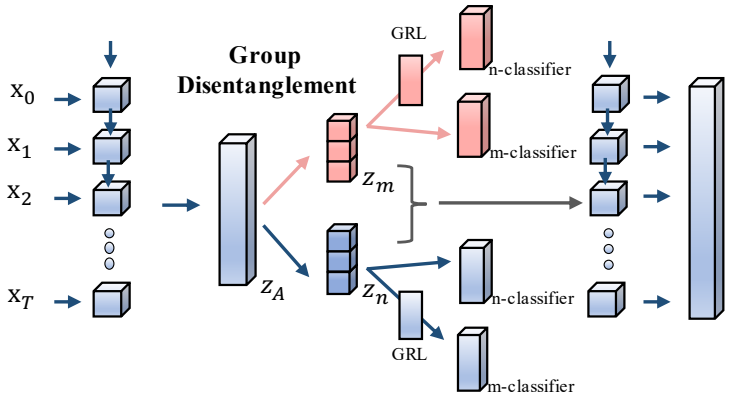


Figure 4.4: The architecture of Latent Group Segment Disentanglement experiment.

maximization term α is set to be 10. Instead of fixing the adaptation factor in the gradient reversal layers (GRL), we gradually change it from 0 to 1 using the following schedule: $\lambda_p = \frac{2}{1+\exp(-\gamma \cdot p)} - 1$, where γ is set to be 10, where p is the training progress linearly changing from 0 to 1.

- **Optimizer:** We train the model by utilizing a batch size of 64 and a momentum of 0.9 with the ADAM optimizer. The learning rate starts at $5e-4$, and is dropped by a factor of 10 at 50% and 75% of the training progress, respectively. The number of epochs has been set to be 200. The exactly same optimizer is also adopted in the baselines.

4.4.2 Performance Evaluation

To answer the research question **Q1**, we apply `DTS` to domain adaptation tasks to validate the *effectiveness of the group segmentation* (Section 2.4), and compare `DTS` with the state-of-the-art algorithms. During the training phase, only the training dataset is accessible: labeled data as the source domain and unlabeled data as the target domain. During the evaluation, the test data becomes available as the target domain for quantitative evaluations and visualizations.

4.4.2.1 Quantitative Results

Table 4.1 compares the performance of `DTS` and the baselines on HAR, HHAR, WISDM AR, and uWave datasets. We include no adaptation as an approximate lower bound, and models trained

Problem	W/O	R-DANN	VRADA	CoDATS	DTS	Target
HAR 2 → 11	83.3	80.7	64.1	74.5	**84.3**	100.0
HAR 7 → 13	89.9	75.3	78.3	96.5	**98.1**	100.0
HAR 12 → 16	41.9	35.1	61.7	77.5	**72.9**	100.0
HAR 12 → 18	90.0	74.9	74.4	100.0	**100.0**	100.0
HAR 9 → 18	31.1	56.6	59.8	85.8	**89.8**	100.0
HAR 14 → 19	62.0	71.3	64.4	98.6	**100.0**	100.0
HAR 18 → 23	89.3	78.2	72.9	89.3	**94.9**	100.0
HAR 6 → 23	52.9	79.1	78.2	94.2	**94.9**	100.0
HAR 7 → 24	94.4	84.8	93.9	99.1	**100.0**	100.0
HAR 17 → 25	57.3	66.3	52.0	97.6	**100.0**	100.0
HAR Average	69.2	70.2	70.0	90.2	**93.5**	100.0
HHAR 1 → 3	77.8	85.1	81.3	90.8	**93.7**	99.2
HHAR 3 → 5	68.8	85.4	82.3	94.3	**95.9**	99.0
HHAR 4 → 5	60.4	70.4	71.6	94.2	**94.9**	99.0
HHAR 0 → 6	33.6	33.4	35.6	76.7	**80.0**	98.8
HHAR 1 → 6	72.1	81.7	74.9	90.8	**92.1**	98.8
HHAR 4 → 6	48.0	64.6	62.7	85.3	**92.3**	98.8
HHAR 5 → 6	65.1	54.4	60.0	91.7	**92.5**	98.8
HHAR 2 → 7	49.4	46.4	45.0	58.1	**64.7**	98.5
HHAR 3 → 8	77.8	82.8	82.2	93.4	**94.7**	99.3
HHAR 5 → 8	95.3	82.5	87.5	95.8	**97.9**	99.3
HHAR Average	64.8	68.7	68.3	86.8	**89.9**	99.0
WISDM 1 → 11	71.7	55.6	55.0	**93.3**	89.6	98.3
WISDM 3 → 11	6.7	28.9	45.0	47.8	**58.3**	98.3
WISDM 4 → 15	78.2	69.2	82.7	81.4	**82.9**	100.0
WISDM 2 → 25	81.1	57.8	72.2	90.6	**95.8**	100.0
WISDM 25 → 29	47.1	61.6	81.9	74.6	**82.2**	95.7
WISDM 7 → 30	62.5	41.7	61.9	73.2	**89.2**	100.0
WISDM 21 → 31	57.1	61.0	68.6	92.4	**96.4**	97.1
WISDM 2 → 32	60.1	49.0	66.7	68.6	**70.7**	100.0
WISDM 1 → 7	68.5	44.8	63.0	66.1	**72.7**	96.4
WISDM 0 → 8	34.7	13.3	14.7	62.0	**77.4**	99.3
WISDM Average	56.8	48.3	61.2	75.8	**81.7**	98.5
uWave 2 → 5	86.3	33.3	18.5	98.2	**100.0**	100.0
uWave 3 → 5	82.7	63.7	32.4	92.9	**95.6**	100.0
uWave 4 → 5	83.3	35.4	12.8	99.1	**96.7**	100.0
uWave 2 → 6	86.0	34.5	25.3	93.8	**97.8**	100.0
uWave 1 → 7	95.2	26.8	29.2	**98.5**	94.4	100.0
uWave 2 → 7	85.1	53.9	12.2	91.4	**98.9**	100.0
uWave 3 → 7	95.5	64.0	30.4	92.0	**98.9**	100.0
uWave 1 → 8	100.0	78.6	11.0	93.8	**100.0**	100.0
uWave 4 → 8	**100.0**	44.0	12.5	96.7	97.8	100.0
uWave 7 → 8	95.2	49.7	12.5	93.8	**96.7**	100.0
uWave Average	91.0	48.4	19.7	94.3	**97.7**	100.0

Table 4.1: Target classification accuracy (based on the class-dependent representation) for time-series domain adaptation.

Dataset	**Acc/D-S**	**Acc/D-T**	**Acc/C-S**	**Acc/C-T**	**AUC/C-S**	**AUC/C-T**
HAR	100.0	100.0	100.0	97.2	100	99.2
HAR/r	54.7	66.7	35.9	22.2	54.6	53.7
HHAR	96.9	98.8	97.9	91.1	99.8	98.7
HHAR/r	70.8	70.9	30.5	17.6	58.3	52.7
WISDM	100	100	94.5	70.7	99.0	85.9
WISDM/r	67.3	46.7	38.2	33.3	69.7	67.9
uWave	100	100	99.1	94.4	99.0	85.9
uWave/r	55.4	48.2	18.8	12.5	54.6	53.7

Table 4.2: Ablation studies of the discriminability.

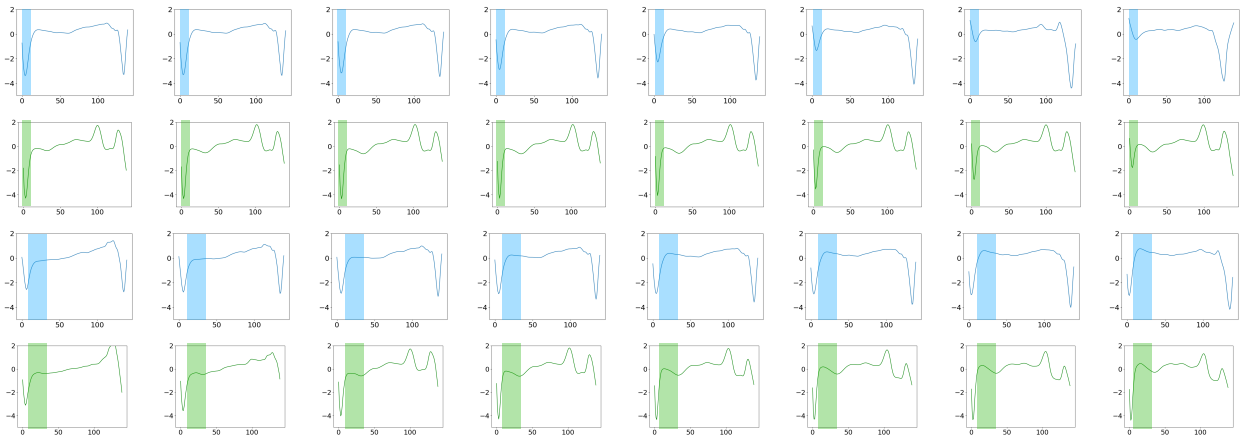


Figure 4.5: Latent traversal plots from DTS on ECG.

directly on labeled target data as upper bound. After the group disentanglement, two variables are used to encode the domain information and the class information, respectively. We use the class-dependent latent variable for classification since it is domain-invariant. We observe that DTS outperforms the baselines with consistently $+3\%$ higher accuracy over all datasets. These results ascertain the effectiveness of DTS in boosting the performance of domain adaptation by obtaining domain-invariant transferable components common knowledge. The enhanced results also validate eliminating irrelevant information from group disentanglement could prevent negative transferring.

4.4.2.2 Ablation Studies

We study whether DTS can decompose the representations into domain-dependent and class-dependent components with a series of ablation studies. We compare the discriminability of the disentangled features, including domain-dependent ($\mathbf{z}_d$) and class-dependent ($\mathbf{z}_y$) components (see Section 2.4.2), sampled from both source and target domains. The ablations include (i) using domain-dependent features $\mathbf{z}_d$ and class-dependent features $\mathbf{z}_y$, and (ii) domain-invariant and class-invariant features (shown as /r), respectively. The comparison between DTS and the ablations (/r) is shown in Table 5.5. We observe that DTS significantly outperforms the ablations over all datasets. Disentangled task-dependent group segments consistently help to improve the performance. Conversely, the class-dependent features are invariant to the change of domains, and the domain-dependent features are invariant to the change of classes. DTS could preserve the discriminability of the disentangled features corresponding to the specified task, and simultaneously make disentangled features less discriminative for the irrelevant task. It indicates that these disentangled group segments do not contain any useful semantic concepts for other irrelevant tasks.

4.4.3 Individual Latent Factor Disentanglement

To answer the research question **Q2**, we provide latent traversals as qualitative results to validate that DTS tends to consistently discover more informative latent factors and provide more meaningful disentangled representations of time-series (see Section 2.3).

4.4.3.1 Traversal Plots to Discover Semantics

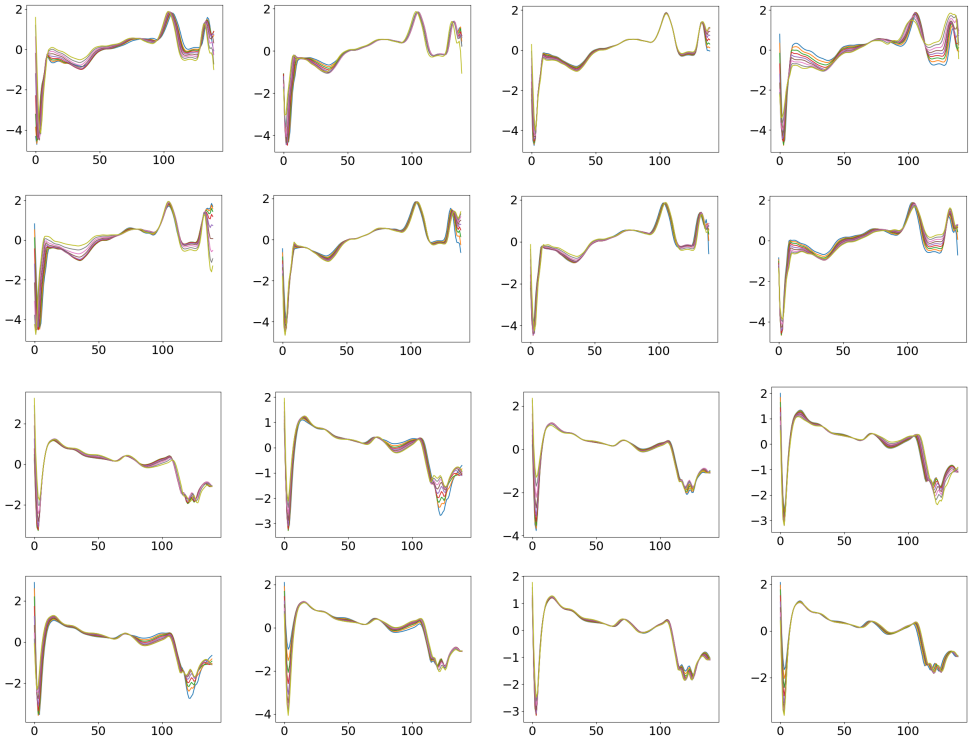
There is currently no general method for quantifying the degree of learned disentanglement or optimize the hyperparameters (unless there are concept ground-truth factors v available, then the mutual information gap (MIG) [62, 61] could be used to determine if there exists a deterministic, invertible relationship between z and v). Therefore, there is no way to quantitatively compare the degree of disentanglement achieved by different models or when optimizing the hyperparameters of a single model. Fig. 4.5 plots the manipulation results of the latent traversal results from DTS. Each block of the figure corresponds to the traversal of a single latent variable while keeping others

fixed. Each row represents a different seed image used to infer the latent values with traversal over the [-4, 4] range. The results show that our manipulation approach performs well on all attributes in both positive and negative directions. We observe that moving the latent codes can produce continuous change, with the sequential patterns orthogonal to the others. According to the editing process, the first and the third rows (sampled from two different time-series of ECG) denote the decline degree at the first turning point, transition from rigid to a more mild manner; the second and the last rows (with the same sampling strategy) denote the rising trend at the second turning point, transition from inconspicuous to obvious. It demonstrates that DTS discovered latent factors in an unsupervised manner that encode sequential trend and depict an interpretable property in the generation. These observations provide strong evidence that DTS does not produce time-series randomly, but learns some interpretable semantics in the latent space.

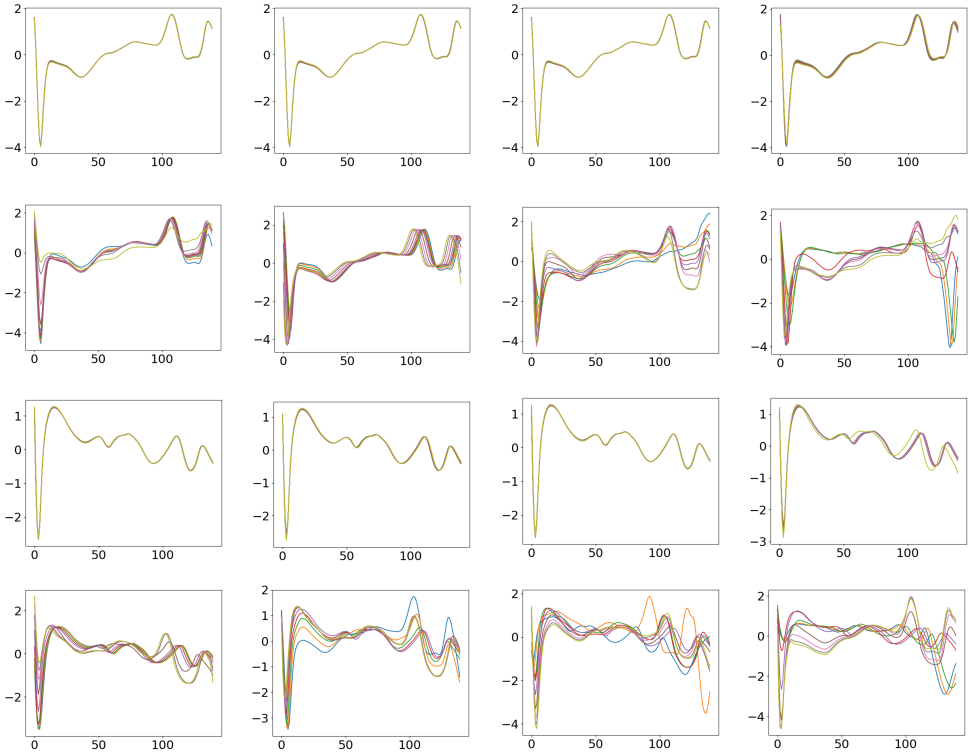
4.4.3.2 Qualitative Comparison of Latent Codes

We train DTS on ECG data to evaluate disentanglement performance for individual latent factors. We use the same traversal way to show the disentanglement quality. Fig. 4.6 provides a qualitative comparison of the disentanglement performance of DTS and LSTM-VAE. We edit the time-series by altering the latent codes in the $\mathcal{Z}$ space. Here, the dimension of the representation is set to be 12. This setting helps reduce the impact of differences in complexity by model frameworks. However, for a better comparison, we only select eight dimensions that change more regularly. The sequences visualized in panels are generated from $Z \sim q\left(Z \mid \boldsymbol{x}_{1:T}\right)$. Hence, the dynamics are imposed by the encoder, but the identity is sampled from the prior.

Fig. 4.6 shows traversals in latent variables that depict an interpretable property in generating time-series. Often it could generate more semantic convincing time-series than LSTM-VAE. Comparing the visualization from panel (a) and panel (b), DTS could generate a time-series in a more diverse way. It can be seen that sampling from an entangled representation results in LSTM-VAE (panel (a)) only reflects small differences according to the traversal perturbations. One possible reason is that the LSTM-VAE is dominated by the reconstruction term during the training phase. The slight changes only correspond to the reconstruction distortion due to the latent codes and



(a) Visualizing the generated time-series from LSTM-VAE.



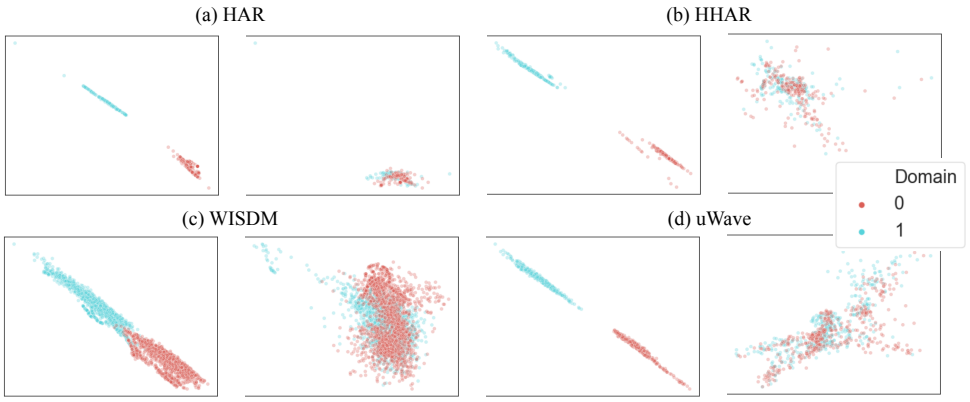
(b) Visualizing the generated time-series from DTS.

Figure 4.6: Comparison of learned latent variables.

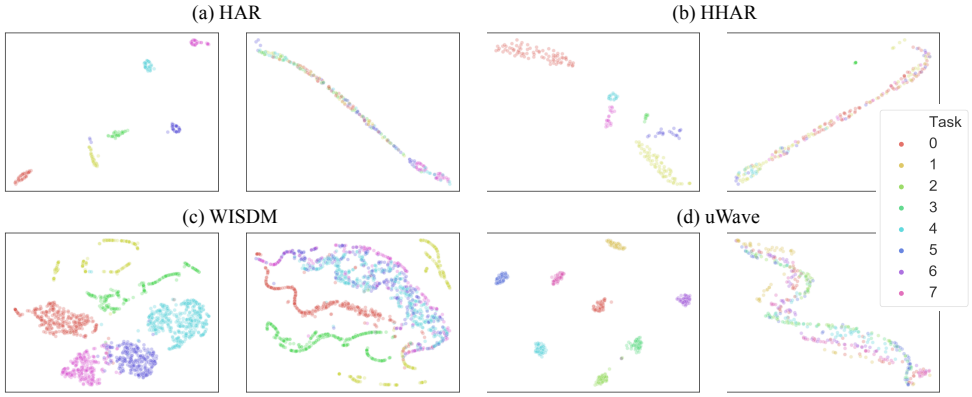
observations are independent. Comparing with LSTM-VAE, some of the DTS latent codes tend to learn a smooth continuous transformation over a wider range of factor values as vibrations. A clear transition process can be observed from the manipulation results with respect to the $\mathbb{Z}$ space. As the value of individual latent factor increases, the semantic of the latent factor changes across different sequential patterns. Other latent codes are robust with the vibrations, as it does not play any role in the generation process. Single latent units are sensitive to changes in single generative factor, while being relatively invariant to changes in other factors. One possible reason is that the representation of the time-series could be effectively expressed with a few latent codes in the $\mathbb{Z}$. Individual latent factor disentanglement process may help us to recognize the useful latent codes, and discard the redundant parts. All these results demonstrate that DTS is able to disentangle useful knowledge from sequential data, which is more informative as interpretable factors in the latent space.

4.4.4 Latent Group Segment Disentanglement

To answer the research question **Q3**, we visualize the disentangled segments over the representation space (see Section 2.4.1). Fig. 4.7(a) shows the effect of domain-dependent and domain-invariant disentanglement on the distribution of the extracted features. We observe that, for all the datasets, the adaptation in DTS makes the disentangled domain(class)-dependent features more distinguishable, but the domain(class)-invariant features indistinguishable. The results validate the DTS can learn decomposed segments that contain independent semantic information. Furthermore, we can observe an apparent clustering effect (the different colors denote different categories). A widely-accepted assumption [85] indicates that observation distribution contains separated data clusters and data samples in the same cluster share the same class label in domain adaptations. These results validate the discriminative ability of the disentanglement, since DTS is capable of yielding strong clustering in the target domain. And it almost matches the prior perfectly, as the semantically similar observations are mapped closer, and create clusters. This phenomenon gives us another insight as the disentangled group segments could enhance the interpretability.



(a) T-SNE visualizations of the DTS activations on the distribution of domain-dependent representation (left) and domain-invariant representations(right). Blue points correspond to the source domain examples, while red ones correspond to the target domain ones.



(b) T-SNE visualizations of DTS activations on the distribution of class-dependent representation (left) and class-invariant representations (right). Each color denotes one specific class.

Figure 4.7: The effect of (a) domain-dependent and -invariant (b) class-dependent and -invariant disentanglement on the distribution of the extracted features.

5. AUTOMATED OUTLIER DETECTION[1]

5.1 Motivation

One drawback of existing deep learning based outlier detection algorithms is that the design of neural architectures heavily relies on human experience to fine-tune the hyperparameters, which is usually time-consuming and may result in sub-optimal performance. In addition, we often need to find a suitable definition of the outlier and its corresponding objective function to differentiate between normal and anomalous behaviors. One common way to define the outliers is to estimate the relative density of each sample, and declare instances that lie in a neighborhood with low density as anomalies [16]. Yet these density-based techniques perform poorly if the data have regions of varying densities. Another way to define anomalies is through clustering. An instance will be classified as normal data if it is close to the existing clusters, while the anomalies are assumed to be far away from any existing clusters [17]. However, these clustering-based techniques will be less effective if the anomalies form significant clusters among themselves [1]. Thus, the proper definition of outliers not only requires domain knowledge from researchers and experience from data scientists, but also needs thorough and detailed raw data analysis efforts.

While Neural Architecture Search (NAS) has achieved competitive performance on supervised image and text classification tasks [86, 87], the objective of existing approaches is limited to the search of neural architectures, which is not suitable for outlier detection systems. Therefore, there exists a vital need for the methods that can automatize the process of finding an effective outlier detection model for a target dataset, which covers not only the architecture settings and hyper-parameters, but also the outlier definitions and the corresponding objective functions.

The challenges of developing an automated outlier detection scheme are three-fold: (1) Lack

[1]This chapter is reprinted with permission from "AutoOD: Neural Architecture Search for Outlier Detection", by Yuening Li, Zhengzhang Chen, Daochen Zha, Kaixiong Zhou, Haifeng Jin, Haifeng Chen, Xia Hu, 2021. Proceedings of the IEEE 37th International Conference on Data Engineering (ICDE). Copyright 2021 by IEEE. This chapter is also reprinted with permission from "Automated Anomaly Detection via Curiosity-Guided Search and Self-Imitation Learning", by Yuening Li, Zhengzhang Chen, Daochen Zha, Kaixiong Zhou, Haifeng Jin, Haifeng Chen, Xia Hu, 2021. IEEE Transactions on Neural Networks and Learning Systems, 2021. Copyright 2021 by IEEE.

of intrinsic search space. It is non-trivial to determine the search space for an outlier detection task. As we mentioned above, different from the search spaces defined by NAS, the search space of automated outlier detection needs to cover not only the architecture configurations, but also the outlier definitions with corresponding objective functions. (2) Unstable search process. The search process may easily become unstable and fragile when outlier detection compounds with architecture search. On the one hand, the unsupervised nature and imbalanced data distributions make the search process easily fall into the local optima [88]. On the other hand, the internal mechanisms of the traditional NAS may introduce bias in the search process. For instance, the weight sharing mechanism makes the architectures who have better initial performance with similar structures more likely to be sampled [89], which leads to misjudgments of the child model's performance. (3) Insufficient abnormal/negative data. In real-world outlier detection tasks, outliers or abnormal samples are very rare. Thus, it requires the search strategy to exploit samples and historical search experiences in a more effective way.

5.2 Preliminaries and Problem Formulation

In this section, we present the preliminaries and problem definition of our work.

5.2.1 Deep AutoEncoder Based Outlier Detection

Classical outlier detection methods, such as Local Outlier Factor and One-Class SVMs, suffer from bad computational scalability and the curse of dimensionality [8]. To make such shallow methods work in high-dimensional, data-rich scenarios, it often requires substantial feature engineering processes. Deep structured models provide a more efficient way to process features. Among recent deep structured studies, Deep AutoEncoders are one of the most promising approaches for outlier detection. The aim of an AutoEncoder is to learn a representation by minimizing the reconstruction error from normal samples [90]. Therefore, these networks are able to extract the common factors of variation from normal samples and reconstruct them easily, and vice versa. Besides directly employing the reconstruction error as the denoter, recent work [17, 16, 8] demonstrate the effectiveness of collaborating Deep AutoEncoders with classical outlier detection

techniques, by introducing regularizers through plugging learned representations into classical outlier definition hypotheses. To be more specific, there are three common outlier assumptions: density, cluster, and centroid. The density based approaches [16] estimate the relative density of each sample, and declare instances that lie in a neighborhood with low density as anomalies. Under the clustering based assumption, normal instances belong to an existing cluster in the dataset, while anomalies are not contained in any existing cluster [17]. The centroid based approaches [8] rely on the assumption that normal data instances lie close to their closest cluster centroid, while anomalies are far away from them. In this work, we illustrate the proposed `AutoOD` by utilizing Deep AutoEncoder with a variety of regularizers as the basic outlier detection algorithm. The framework of `AutoOD` could be easily extended to other deep-structured outlier detection approaches.

5.2.2 Related Work

In this section, we review the related work on Neural Architecture Search (NAS). Recently, NAS has attracted increasing research interests. Its goal is to find the optimal neural architecture in a predefined search space to maximize the model performance on a given task. Designing a NAS algorithm requires two key components: the search space and the search strategy (optimization algorithm) [91].

The search space defines which architectures can be represented in principles. The existing work of search space follows two trends: the macro and micro search [86, 92]. The macro search provides an exhaustive-architecture search space to encourage the controller to explore the space and discover novel architectures, while the micro search inductively limits the search space to accelerate the search process. The choice and the size of the search space determine the difficulty of the optimization problem. Yet, even for the case of the search space based on a single cell, it is still a challenging problem due to the discrete search space and the curse of high-dimensionality (since more complex models tend to perform better, resulting in more design choices) [91]. Thereby, incorporating prior knowledge about the typical properties of architectures well-suited for a task can significantly reduce the size of the search space and simplify the search process. Recent research [93] has validated the importance of the search space in the search process. With exten-

sive experimental reproducibility studies, a task-tailored, carefully-designed search space plays a more important role than the other search strategies. Recent works have proposed tailored search spaces with their applications, including image segmentation [94], adversarial training [95], and augmentation strategies [96]. To the best of our knowledge, our proposed `AutoOD` describes the first attempt to design the search space specifically customized to the anomaly detection task. `AutoOD` uses the micro search space to keep consistent with previous works. Yet the contribution of `AutoOD` in search space is to design a hierarchical, general-purpose search space, including global settings for the whole model, and local settings in each layer independently. Moreover, our proposed search space not only covers the hyperparameters as architecture configurations, such as the size of convolutional kernels and filters in each layer, but also incorporates the definition-hypothesis and its corresponding objective function.

The search strategy focuses on how to explore the search space. Recent approaches include reinforcement learning (RL) [87, 97], Bayesian optimization [98], and gradient-based methods [99, 100, 101]. Although these methods have improved upon human-designed architectures, directly borrowing existing NAS ideas from image classification to anomaly detection will not work. Due to the imbalanced data, the search process becomes more unstable in anomaly detection tasks [88]. As an internal mechanism in the traditional NAS, weight sharing, also introduces the inductive bias in the search process which intensifies the tendency [89]. Weight sharing [86] is proposed to transfer the well-trained weight before to a sampled architecture, to avoid training the offspring architecture from scratch. Recent research has validated that the weight sharing mechanism makes the architectures who have better initial performance with similar structures more likely to be sampled [89], which leads to misjudgments of the child model's performance. Our work builds upon RL-based method, which uses a recurrent neural network controller to choose blocks from its search space. Beyond that, we propose a curiosity-guided search strategy to stabilize the search process via encouraging the controller to seek out unexplored regions in the search space. Our search strategy formulates the search process as a classical exploration-exploitation trade-off. On one hand, it is encouraged to find the optimal child model more efficiently; on the other hand, it

avoids the premature convergence to a sub-optimal region due to the inductive bias or insufficient search.

Previous work has explored reinforcement learning (RL) in the context of anomaly detection. Oh *et al.* [102] formulate sequential anomaly detection as an inverse RL problem, where the reward function is inferred from the behavior data. Pang *et al.* [103] propose a deep RL approach to actively explore novel anomaly classes in semi-supervised settings. Zha *et al.* [104] use deep RL to meta-learn an active learning strategy. However, these studies mainly focus on identifying anomalies with RL but do not consider neural architectures. In this work, we use RL to search the optimal neural architectures, which complements previous studies.

5.2.3 Problem Statement

Different from the traditional Neural Architecture Search, which focuses on optimizing neural network architectures, automated outlier detection has the following two unique characteristics. First, the neural architecture in the Autoencoder needs to be adaptive in the given dataset to achieve competitive performance. The hyperparameter configurations of neural architecture include the number of layers, the size of convolutional kernels and filters, etc.; Second, the outlier detection requires the designs of the definition-hypothesis and corresponding objective function. Formally, we define the outlier detection model and the unified optimization problem of automated outlier detection as follows.

Theorem 7. ***Outlier Detection Model****: The model of outlier detection consists of three key components: the neural network architecture A of AutoEncoder, the definition-hypothesis H of outlier assumption, and the loss function L. We represent the model as a triple (A, H, L).*

Theorem 8. ***Automated Outlier Detection****: Let the triple $(\mathcal{A}, \mathcal{H}, \mathcal{L})$ denote the search space of outlier detection models, where $\mathcal{A}$ denotes the architecture subspace, $\mathcal{H}$ denotes the definition-hypothesis subspace, and $\mathcal{L}$ denotes the loss functions subspace. Given training set $\mathcal{D}_{train}$ and validation set $\mathcal{D}_{valid}$, we aim to find the optimal model $(A^\star, H^\star, L^\star)$ to minimize the objective*

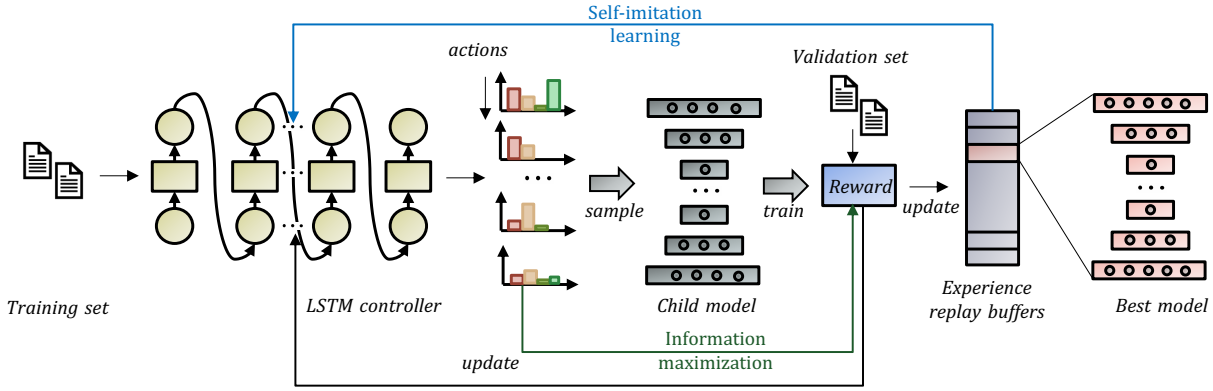


Figure 5.1: An overview of AutoOD .

function $\mathcal{J}$ as follows:

$$(A^\star, H^\star, L^\star) = \underset{A\in\mathcal{A}, H\in\mathcal{H}, L\in\mathcal{L}}{\operatorname{argmin}} \mathcal{J}(A(\omega), H, L, \mathcal{D}_{train}, \mathcal{D}_{valid}), \tag{5.1}$$

where ω denotes the weights that are well trained on architecture A. $\mathcal{J}$ denotes the loss on $\mathcal{D}_{valid}$ using the model trained on the $\mathcal{D}_{train}$ with definition-hypothesis H and loss function L.

5.3 AutoOD Framework

In this section, we propose an automated outlier detection framework to find the optimal neural network model for a given dataset. A general search space is designed to include the neural architecture hyperparameters, definition-hypothesis, and objective function. To overcome the curse of local optimality under certain unstable search circumstances, we propose a curiosity-guided search strategy to improve search effectiveness. Moreover, we introduce an experience replay mechanism based on self-imitation learning to better exploit the past good experiences and enhance the sample efficiency. An overview of AutoOD is given in Figure 5.1.

5.3.1 Search Space Design

Because there is a lack of intrinsic search space for outlier detection tasks, here we design the search space for the Deep AutoEncoder based algorithms, which is composed of global settings

for the whole model, and local settings in each layer independently. Formally, we have:

$$
\begin{aligned}
&A = \{f^1(\cdot), ..., f^N(\cdot), g^1(\cdot), ..., g^N(\cdot)\}, \\
&f^i(x;\omega_i) = \text{ACT}(\text{NORMA}(\text{POOL}(\text{CONV}(x))), \\
&g^i(x;\omega_i) = \text{ACT}(\text{NORMA}(\text{UPPOOL}(\text{DECONV}(f(x))))), \\
&\text{score} = \text{DIST}(g(f(x;\omega)), x) + \text{DEFINEREG}(f(x;\omega)),
\end{aligned}
\tag{5.2}
$$

where x denotes the set of instances as input data, and ω denotes the trainable weight matrix. The architecture space A contains N encoder-decoder layers. $f(\cdot)$ and $g(\cdot)$ denote encoder and decoder functions, respectively. ACT$(\cdot)$ is the activation function set. NORMA denotes the normalization functions. POOL$(\cdot)$ and UPPOOL$(\cdot)$ are pooling methods. CONV$(\cdot)$ and DECONV$(\cdot)$ are convolution functions. DIST$(\cdot)$ is the metric to measure the distance between the original inputs and the reconstruction results. DEFINEREG$(\cdot)$ acts as an regularizer to introduce the definition-hypothesis from H. We revisit and extract the outlier detection hypotheses and their mathematical formulas from state-of-the-art approaches as shown in the Table 5.1. We decompose the search space defined in Eq. (5.2) into the following 8 classes of actions:

Global Settings:

- `Definition-hypothesis` determines the way to define the "outliers" from a high-level perspective, including density based, cluster based, centroid based, and reconstruction based assumptions.

- `Distance measurement` stands for the matrix to measure the distance for the reconstruction purpose, including l_1 norm, l_2 norm, $l_{2,1}$ norm, and the structural similarity (SSIM).

Local Settings in Each Layer:

- `Output channel` is the number of channels produced by the convolution operations in each layer, *i.e.*, 3, 8, 16, 32, 64, 128, 256.

Definitions H	Regularizer Equations
Density [16]	$-\log\left(\sum_{k=1}^{K}\hat{\phi}_k\frac{\exp\left(-\frac{1}{2}(f(x_i;\omega)-\hat{\mu}_k)^T\hat{\mathbf{\Sigma}}_k^{-1}(f(x_i;\omega)-\hat{\mu}_k)\right)}{\sqrt{\left\|2\pi\hat{\mathbf{\Sigma}}_k\right\|}}\right)$
Cluster [17]	$\sum_i\sum_j p_{ij}\log p_{ij}\left(\frac{\left(1+\|f(x_i;\omega)-\mu_j\|^2\right)^{-1}}{\sum_j\left(1+\|f(x_i;\omega)-\mu_j\|^2\right)^{-1}}\right)^{-1}$
Centroid [8]	$R^2+\sum_{i=1}^{n}\max\left\{0,\|f\left(x_i;\omega\right)-c\|^2-R^2\right\}$
Reconstruction [90]	$\frac{1}{n}\sum_{i=1}^{n}\|g\left(f\left(x_i;\omega\right)\right)-x_i\|_2^2$

Table 5.1: The set of four representative outlier detection hypotheses, where $f(\cdot)$ and $g(\cdot)$ denote encoder and decoder functions, respectively.

- `Convolution kernel` denotes the size of the kernel produced by the convolution operations in each layer, *i.e.*, $1\times1, 3\times3, 5\times5, 7\times7$.
- `Pooling type` denotes the type of pooling in each layer, including the max pooling and the average pooling.
- `Pooling kernel` denotes the kernel size produced by the pooling operations in each layer, *i.e.*, $1\times1, 3\times3, 5\times5, 7\times7$.
- `Normalization type` denotes the normalization type in each layer, including three options: batch normalization, instance normalization, and no normalization.
- `Activation function` is a set of activation functions in each layer, including Sigmoid, Tanh, ReLU, Linear, Softplus, LeakyReLU, ReLU6, and ELU.

Thus, we use a $(6N+2)$ element tuple to represent the model, where N is the number of layers in the encoder-decoder-wise structure. Our search space includes an exponential number of settings. Specifically, if the encoder-decoder cell has N layers and we allow action classes as above, it provides $4\times4\times(7\times4\times2\times4\times3\times8)^N$ possible settings. Suppose we have a $N=6$, the number of points in our search space is $3.9e+23$, which requires an efficient search strategy to find an optimal model out of the large search space. Fig. 5.2 illustrates an example of the proposed search space in AutoOD.

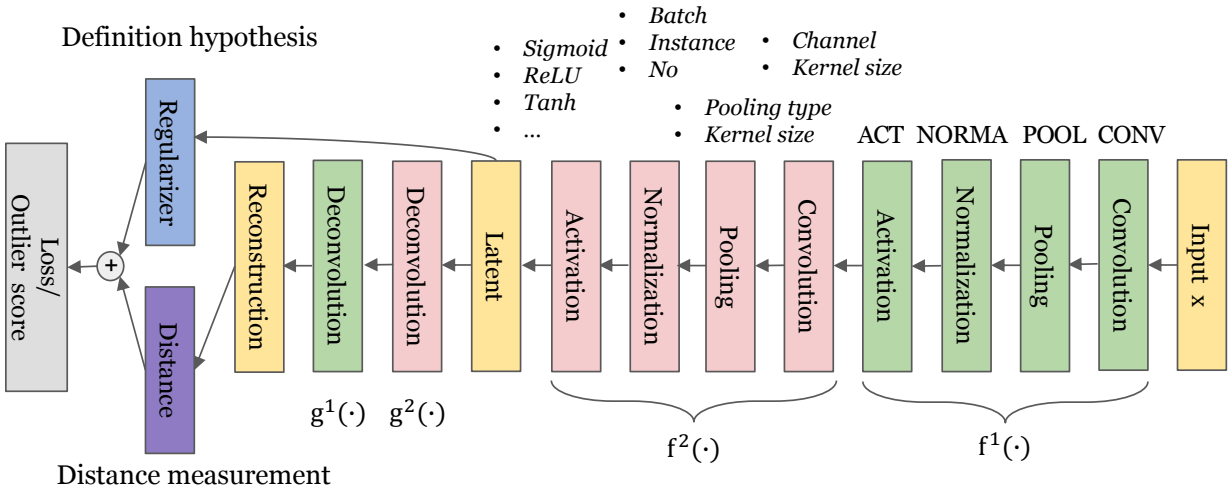


Figure 5.2: An example of the search space in AutoOD with two layers, which is composed of global settings for the whole model and local settings in each layer.

5.3.2 Curiosity-guided Search

We now describe how to search the optimal model within the given search space. Inspired by the recent NAS work, the search strategy is considered as a meta-learning process. A controller is introduced to explore a given search space by training a child model to get an evaluation for guiding exploration [86]. The controller is implemented as a recurrent neural network. We use the controller to generate a sequence of actions for the child model. The whole process can be treated as a reinforcement learning problem with an action $a_{1:T}$, and a reward function r. To find the optimal model, we ask our controller to maximize its expected reward r, which is the expected performance in the validation set of the child models.

There are two sets of learnable parameters: one of them is the shared parameters of the child models, denoted by ω, and the other one is from the LSTM controller, denoted by θ. ω is optimized using stochastic gradient descent (SGD) with the gradient ∇_ω as:

$$\nabla_\omega \mathbb{E}_{m\sim\pi(m;\theta)}[L(m;\omega)] \approx \nabla_\omega L(m,\omega), \tag{5.3}$$

where child model m is sampled from the controllers actions $\pi(m;\theta)$, $L(m,\omega)$ is the loss function composed from the search space above, computed on a minibatch of training data. The gradient is estimated using the Monte Carlo method.

Since the reward signal r is non-differentiable, to maximize the expected reward r, we fix ω and apply the REINFORCE rule [105] to update the controller's parameters θ as:

$$\nabla_\theta \mathbb{E}_{P(a_{1:t};\theta)}[r\nabla_\theta \log P(a_t|a_{1:t-1};\theta)], \tag{5.4}$$

where r is computed as the performance on the validation set, rather than on the label-free training set. We define the reward r as the detection accuracy of the sampled child model. We also adopt different evaluation metrics, including AUROC, AUPR and RPRO in the following experiment section. An empirical approximation of the Eq. (5.4) is:

$$L = \frac{1}{n}\sum_{k=1}^{n}\sum_{t=1}^{T}(r_k - b)\nabla_\theta \log P(a_t|a_{1:t-1};\theta), \tag{5.5}$$

where n is the number of different child models that the controller samples in one batch and T is the number of tokens. b acts as a baseline function to reduce the variance of this estimate.

5.3.2.1 Curiosity-driven Exploration

Despite being widely utilized due to search efficiency, weight sharing approaches are roughly built on empirical experiments instead of solid theoretical ground [89]. The unfair bias will make the controller misjudge the child-model performance: those who have better initial performance with similar child models are more likely to be sampled. In the meanwhile, due to the imbalanced label distribution in outlier detection tasks, it is easy to make the controller fall into local optima.

To address these problems, `AutoOD` builds on the theory of curiosity-driven exploration [106], aiming to encourage the controller to seek out regions in the searching spaces that are relatively unexplored. It brings us a typical exploration-exploitation dilemma to guide the controller.

Bayesian reinforcement learning [107, 108] offers us a formal guarantees as coherent proba-

bilistic model for reinforcement learning. It provides a principled framework to express the classic exploration-exploitation dilemma, by keeping an explicit representation of uncertainty, and selecting actions that are optimal with respect to a version of the problem that incorporates this uncertainty [108]. Here, instead of a vanilla RNN, we use a Bayesian LSTM as the structure of the controller to guide the search. The controllers understanding of the search space is represented dynamically over the uncertainty of the parameters of the controller. Assuming a prior $p(\theta)$, it maintains a distribution prior over the controller's parameters through a distribution over θ. The controller models the actions via $p(a_t|a_{1:t};\theta)$, paratetrized by θ. According to curiosity-driven exploration [109], the uncertainty about the dynamics of the controller can be formalized as maximizing the information:

$$I(a_t;\theta|a_{1:t-1}) = \mathbb{E}_{a_t\sim P(\cdot|a_{1:t-1})}\big[D_{\mathrm{KL}}[p(\theta|a_{1:t-1})\,||\,p(\theta)]\big], \tag{5.6}$$

where the KL divergence can be interpreted as *information gain*, which denotes the mutual information between the controller's new belief over the model to the old one.

Thus, the information gain of the posterior dynamics distribution of the controller can be approximated as an *intrinsic reward*, which captures the controllers surprise in the form of a reward function. We can also use the REINFORCE rule to approximate planning for maximal mutual information by adding the intrinsic reward along with the external reward (accuracy on the validation set) as a new reward function. It can also be interpreted as a trade-off between exploitation and exploration as:

$$r_{\text{new}}(a_t) = r(a_t) + \eta D_{\mathrm{KL}}[p(\theta|a_{1:t-1})\,||\,p(\theta)], \tag{5.7}$$

where $\eta \in \mathbb{R}_+$ is a hyperparameter controlling the urge to explore. However, it is generally intractable to calculate the posterior $p(\theta|a_{1:t-1})$ in Eq. (5.7).

5.3.2.2 Variational Bayes-by-Backprop

In this subsection, we propose a tractable solution to maximize the information gain objective presented in the previous subsection. To learn a probability distribution over network parameters

θ, we propose a practical solution through a back-propagation compatible algorithm, *bayes-by-backprop* [106, 107].

In Bayesian models, latent variables are drawn from a prior density $p(\theta)$. During infernece, the posterior distribution $p(\theta|x)$ is computed given a new action through Bayes' rule as:

$$p(a_t|a_{1:t-1}) = \frac{p(\theta)p(a_t|a_{1:t-1};\theta)}{p(a_t|a_{1:t-1})}. \tag{5.8}$$

The denominator can be computed through the integral:

$$p(a_t|a_{1:t-1}) = \int_{\Theta} p(a_t|a_{1:t-1};\theta)p(\theta)\mathrm{d}\theta. \tag{5.9}$$

As controllers are highly expressive parametrized LSTM networks, which are usually intractable as high-dimensionality. Instead of calculating the posterior $p(\theta|\mathcal{D}_{\text{train}})$ for a training dataset $\mathcal{D}_{\text{train}}$. We approximate the posterior through an alternative probability densities over the latent variables θ as $q(\theta)$, by minimizing the Kullback-Leibler(KL) divergence $D_{\mathrm{KL}}[q(\theta)\,||\,p(\theta)]$. We use $\mathcal{D}$ instead of $\mathcal{D}_{\text{train}}$ in the following parts of this subsection for brevity.

$$q(\theta) = \prod_{i=1}^{|\Phi|} \mathcal{N}(\theta_i|\mu_i;\sigma_i^2). \tag{5.10}$$

$q(\theta)$ is given by a Gaussian distribution, with μ as the Gaussian's mean vector and σ as the covariance matrix.

Once minimized the KL divergence, $q(\cdot)$ would be the closest approximation to the true posterior. Let $\log p(\mathcal{D}|\theta)$ be the log-likelihood of the model. Then, the network can be trained by minimizing the variational free energy as the expected lower bound:

$$L[q(\theta),\mathcal{D}] = -\mathbb{E}_{\theta\sim q(\cdot)}\big[\log p(\mathcal{D}|\theta)\big] + D_{\mathrm{KL}}\big[q(\theta)\,||\,p(\theta)\big], \tag{5.11}$$

which can be approximated using N Monte Carlo samples from the variational posterior with N

samples drawn according to $\theta \sim q(\cdot)$:

$$L[q(\theta), \mathcal{D}] \approx \sum_{i=1}^{N} -\log p(\mathcal{D}|\theta^{(i)}) + \log q(\theta^{(i)}) - \log p(\theta^{(i)}). \tag{5.12}$$

5.3.2.3 Posterior Sharpening

We discuss how to derive a distribution $q(\theta|\mathcal{D})$ to improve the gradient estimates of the intractable likelihood function $p(\mathcal{D})$, which is related to Variational AutoEncoders (VAEs) [66]. Inspired from strong empirical evidence and extensive work on VAEs, the "sharpened" posterior yields more stable optimization. We now use posterior sharpening strategy to benefit our search process.

The challenging part of modelling the variational posterior $q(\theta|\mathcal{D})$ is the large number of dimensions of $\theta \in \mathbb{R}^d$, which makes the modelling unfeasible. Given the first term of the loss $-\log p(\mathcal{D}|\theta)$ is differentiable with respect to θ, we propose to parameterize q as a liniear combination of θ and $-\log p(\mathcal{D}|\theta)$. Thus, we can define the hierarchical posterior of the form in Eq. (5.10):

$$q(\theta|\mathcal{D}) = \int q(\theta|\phi, \mathcal{D}) q(\phi) \mathrm{d}\phi, \tag{5.13}$$

$$q(\theta|\phi, \mathcal{D}) = \mathcal{N}(\theta|\phi - \eta * -\nabla_\phi \log p(\mathcal{D}|\phi), \sigma^2 I) \tag{5.14}$$

with $\mu, \sigma \in \mathbb{R}^d$, and $q(\phi) = \mathcal{N}(\phi|\mu, \sigma)$ as the same setting in the standard variational inference method. $\eta \in \mathbb{R}^d$ can be treated as a per-parameter learning rate.

In the training phrase, we have $\theta \sim q(\theta|\mathcal{D})$ via ancestral sampling to optimise the loss as:

$$L_{\text{explore}} = L(\mu, \sigma, \eta) = \mathbb{E}_{\mathcal{D}}[\mathbb{E}_{q(\phi)q(\theta|\phi,\mathcal{D})}[L(\mathcal{D}, \theta, \phi|\mu, \sigma, \eta)]] \tag{5.15}$$

with $L(\mathcal{D}, \theta, \phi|\mu, \sigma, \eta)$ given by:

$$L(\mathcal{D}, \theta, \phi|\mu, \sigma, \eta) = -\log p(\mathcal{D}|\theta) + \mathrm{KL}[q(\theta|\phi, \mathcal{D}) \,||\, p(\theta|\phi)] + \frac{1}{C}\mathrm{KL}[q(\phi) \,||\, p(\phi)], \tag{5.16}$$

where the constant C is the number of truncated sequences.

Thus, we turn to deriving the training loss function for posterior sharpening. With the discussion above, we assume a hierarchical prior for the parameters such that $p(\mathcal{D}) = \int p(\mathcal{D}|\theta)p(\theta|\phi)p(\phi)\mathrm{d}\theta\mathrm{d}\phi$. Then, the expected lower bound on $p(\mathcal{D})$ is defined as follows:

$$\begin{aligned}
\log p(\mathcal{D}) &= \log\left(\int p(\mathcal{D}|\theta)p(\theta|\phi)p(\phi)\mathrm{d}\theta\mathrm{d}\phi\right) \\
&\geqslant \mathbb{E}_{q(\phi,\theta|\mathcal{D})}\left[\log\frac{p(\mathcal{D}|\theta)p(\theta|\phi)p(\phi)}{q(\phi,\theta|\mathcal{D})}\right] \\
&= \mathbb{E}_{q(\theta|\phi,\mathcal{D})q(\phi)}\left[\log\frac{p(\mathcal{D}|\theta)p(\theta|\phi)p(\phi)}{q(\theta|\phi,\mathcal{D})q(\phi)}\right] \\
&= \mathbb{E}_{q(\phi)}\left[\mathbb{E}_{q(\theta|\phi,\mathcal{D})}\left[\log p(\mathcal{D}|\theta) + \log\frac{p(\theta|\phi)}{q(\theta|\phi,\mathcal{D})}\right] + \log\frac{p(\phi)}{q(\phi)}\right] \\
&= \mathbb{E}_{q(\phi)}\Big[\mathbb{E}_{q(\theta|\phi,\mathcal{D})}\big[\log p(\mathcal{D}|\theta) - \mathrm{KL}[q(\theta|\phi,\mathcal{D})\,||\,p(\theta|\phi)]\big] - \mathrm{KL}[q(\phi)\,||\,p(\phi)]\Big].
\end{aligned} \tag{5.17}$$

5.3.3 Experience Replay via Self-Imitation Learning

The goal of this subsection is to exploit the past good experiences for the controller to benefit the search process by enhancing the sample efficiency, especially considering there are only a limited number of negative samples in outlier detection tasks. In this subsection, we propose to store rewards from historical episodes into experience replay buffers [110]: $\mathcal{B} = (a_{1:t}, r_{\boldsymbol{a}})$, where ($a_{1:t}$ and $r_{\boldsymbol{a}}$) are the actions and the corresponding reward. To exploit good past experiences, we update the experience replay buffer for child models with better rewards, and amplify the contribution from them to the gradient of θ. More specifically, we sample child models from the replay buffer using the clipped advantage $(r-b)_+$, where the rewards r in the past experiences outperform the current baseline b. Comparing with the Eq. (5.4) and Eq. (5.5), the objective to update the controller's parameter θ through the replay buffer is:

$$\nabla_\theta \mathbb{E}_{a_{1:t}\sim\pi_\theta, b\sim\mathcal{B}}[-\log\pi_\theta\left(a_t|a_{1:t-1}\right)(r_{\boldsymbol{a}} - b)_+]. \tag{5.18}$$

Algorithm 3 Automated Outlier Detection

Input: Input datasets $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{valid}}$, and search space $\mathcal{S}$.
Output: Optimal model with the best performance.
Initialize parameter θ, ω;
Initialize replay buffer $\mathcal{B} \leftarrow \emptyset$;
for each iteration **do**
    Perform curiosity-guided search via a LSTM controller
    for each step t **do**
        Sample an action $a_t \sim \pi(a_{1:t-1}; \theta)$;
        $\omega \leftarrow \omega - \eta \nabla_\omega \mathbb{E}_{a_t \sim \pi(a_{1:t-1};\theta)}[L(a_{1:t-1}; \omega)]$; ▷ Eq. (5.3)
        $\theta \leftarrow \theta - \eta L_{\text{explore}}(\mathcal{D}_{\text{train}}, \theta)$; ▷ Eq. (5.15)
        $r_{\text{new}}(a_t) \leftarrow r(a_t) + \eta D_{\text{KL}}[p(\theta|a_{1:t-1}) \,||\, p(\theta)]$; ▷ Eq. (5.7)
        Update controller via the new reward $r_{\text{new}}(a_t)$; ▷ Eq. (5.4)
        if the performance of a_t on $\mathcal{D}_{\text{val}}$ outperforms the actions stored in $\mathcal{B}$ **then**
            $\mathcal{B} \leftarrow \{\boldsymbol{a}, r\} \cup \mathcal{B}$; Update replay buffer;
        end if
    end for
    Perform self-imitation learning
    for each step t **do**
        Sample a mini-batch $\{\boldsymbol{a}, r\}$ from $\mathcal{B}$;
        $\omega \leftarrow \omega - \eta \nabla_\omega \mathbb{E}_{a_t \sim \pi(a_{1:t-1};\theta)}[L(a_{1:t-1}; \omega)]$; ▷ Eq. (5.3)
        $\theta \leftarrow \theta - \eta L_{\text{replay}}(\mathcal{D}_{\text{valid}}, \theta)$; ▷ Eq. (5.19)
    end for
end for

Then, an empirical approximation of the Eq. (5.18) is:

$$L_{\text{replay}} = \frac{1}{n} \sum_{k=1}^{n} \sum_{t=1}^{T} \nabla_\theta - \log \pi_\theta \left(a_t | a_{1:t-1}\right) \left(r_{\boldsymbol{a}} - b\right)_+, \tag{5.19}$$

where n is the number of different child models that the controller samples in one batch and T is the number of tokens.

Overall, the joint optimization process is specified in Algorithm 3, which consists of two phrases: the curiosity-guided search process and the self-imitation learning process. After getting the optimal model with the best performance on the validation set, we utilize the searched model for the outlier detection tasks.

5.4 Experiments

In this section, we conduct extensive experiments to answer the following four research questions.

- **Q1**: How effective is `AutoOD` compared with **state-of-the-art handcrafted algorithms**?

  Q2: Whether or not the two key components of `AutoOD`, *i.e.*, **curiosity-guided search and experience replay**, are effective in the search process?

- **Q3**: Compared with **random search**, how effective is the proposed search strategy?

- **Q4**: Does `AutoOD` have the potential to be applied in **more complicated real-world applications**?

5.4.1 Datasets and Tasks

We evaluate `AutoOD` on seven benchmark datasets for instance-level abnormal sample detection and pixel-level defect region segmentation tasks. We also conduct a case study on the CAT [111] dataset.

- **MNIST [112]**: An image dataset consists of handwritten digits. It has a training set of $60,000$ examples, and a test set of $10,000$ examples.

- **Fashion-MNIST [113]**: A MNIST-like dataset contains fashion product with a training set of $60,000$ examples and a test set of $10,000$ examples. Each example is a 28×28 grayscale image associated with a label from 10 classes.

- **CIFAR-10 [114]**: A image dataset consists of $50,000$ training images and $10,000$ test images in 10 different classes. Each example is a 32×32 3-channel image.

- **Tiny-ImageNet [115]**: An image dataset consists of a subset of ImageNet images. It contains $10,000$ test images from 200 different classes. We downsample each image to the size of 64×64.

- **MVTec-AD [116]**: A benchmark dataset relates to industrial inspection in the application of anomaly detection. It contains over 5000 high-resolution images divided into fifteen categories in terms of different objects and textures. Each category comprises two parts: a training set of defect-free images, as well as a test set composed of defect-free images and the ones with various defects. We downsample each image to size 224×224.

- **CAT [111]**: A cat dataset includes $10,000$ cat images. We downsample each image to size 224×224.

- **Gaussian Noise**: A synthetic Gaussian noise dataset consists of $1,000$ random 2D images, where the value of each pixel is sampled from an i.i.d Gaussian distribution with mean 0.5 and unit variance. We further clip each pixel into the range $[0, 1]$.

- **Uniform Noise:** A synthetic uniform noise dataset consists of $1,000$ images, at which the value of each pixel is sampled from an i.i.d uniform distribution on $[0, 1]$.

For the instance-level abnormal sample detection task, we use four benchmark datasets (*i.e.*, MNIST [112], Fashion-MNIST [113], CIFAR-10 [114], and Tiny-ImageNet [115]), and two synthetic noise datasets (*i.e.*, Gaussian and Uniform). Synthetic noise datasets consist of $1,000$ random 2D images, where the value of each pixel is sampled from an i.i.d Gaussian distribution with mean 0.5 and unit variance. We further clip each pixel into the range $[0, 1]$, or an i.i.d uniform distribution on $[0, 1]$. Different datasets contain different classes of images. We manually injected abnormal samples (a.k.a. out-of-distribution samples), which consists of images randomly sampled from other datasets. For all the six datasets, we train an anomaly detection model on the training set, which only contains in-distribution samples, and use a validation set with out-of-distribution samples to guide the search, and another test set with out-of-distribution samples to evaluate the performance. The contamination ratio in the validation set and the test set are both 0.05. The train/validation/test split ratio is $6 : 2 : 2$. Two state-of-the-art methods including MSP [117] and ODIN [118] are used as baselines.

For the pixel-level defect region segmentation task, we use a real-world dataset MVTec-AD [116]. MVTec-AD contains high-resolution images with different objects and texture categories. Each category comprises a set of defect-free training images and a test set of images with various kinds of defects and images without defects. We train the model on the defect-free training set, and split the whole test set into two halves for validation and testing. Three state-of-the-art methods including AutoEncoder [119], AnoGAN [120], and Feature Dictionary [121] are used as baselines.

5.4.2 Baselines

We compare `AutoOD` with five state-of-the-art handcrafted algorithms and the random search strategy.

- **MSP [117]**: The softmax probability distribution is used to detect the anomalies in tasks of computer vision, natural language processing, and automatic speech. The anomaly detection is performed based on the following assumption: the correctly classified examples have greater maximum softmax probabilities than those of erroneously classified and out-of-distribution examples.

- **ODIN [118]**: The pre-trained neural network is reused to detect the out-of-distribution images. ODIN separates the softmax probability distributions between in- and out-of-distribution instance, by using temperature scaling and adding small perturbations on the image data.

- **AutoEncoder [119]**: The structure of convolutional AutoEncoders is applied for unsupervised defect segmentation on image data. More specifically, it utilizes the loss function based on structural similarity, and successfully examines inter-dependencies between local image regions to reveal the defective regions.

- **AnoGAN [120]**: It is a deep convolutional generative adversarial network used to identify the anomalous image data. It learns a manifold of normal anatomical variability, and maps images to a latent space to estimate the anomaly scores.

- **Feature Dictionary [121]**: It applies the convolutional neural networks and self-similarity

	Category	AutoOD	AutoEncoder	AnoGAN	Feature Dictionary
Textures	Carpet	**0.69 / 0.92**	0.38 / 0.59	0.34 / 0.54	0.20 / 0.72
	Grid	**0.89 / 0.94**	0.83 / 0.90	0.04 / 0.58	0.02 / 0.59
	Leather	**0.81 / 0.92**	0.67 / 0.75	0.34 / 0.64	0.74 / 0.87
	Tile	**0.26 / 0.94**	0.23 / 0.51	0.08 / 0.50	0.14 / 0.73
	Wood	**0.47 / 0.97**	0.29 / 0.73	0.14 / 0.62	**0.47** / 0.91
Objects	Bottle	**0.33 / 0.93**	0.22 / 0.86	0.05 / 0.86	0.07 / 0.78
	Cable	**0.17 / 0.87**	0.05 / 0.86	0.01 / 0.78	0.13 / 0.79
	Capsule	**0.16 / 0.95**	0.11 / 0.88	0.04 / 0.84	0.00 / 0.84
	Hazelnut	**0.46 / 0.97**	0.41 / 0.95	0.02 / 0.87	0.00 / 0.72
	Metal Nut	**0.30 / 0.88**	0.26 / 0.86	0.00 / 0.76	0.13 / 0.82
	Pill	**0.30 / 0.92**	0.25 / 0.85	0.17 / 0.87	0.00 / 0.68
	Screw	**0.34 / 0.96**	**0.34 / 0.96**	0.01 / 0.80	0.00 / 0.87
	Toothbrush	**0.60 / 0.90**	0.51 / 0.83	0.07 / **0.90**	0.00 / 0.77
	Transistor	**0.23 / 0.96**	0.22 / 0.86	0.08 / 0.80	0.03 / 0.66
	Zipper	**0.20 / 0.88**	0.13 / 0.77	0.01 / 0.78	0.00 / 0.76

Table 5.2: Performance comparison on pixel-level defect region segmentation.

.

to detect and localize anomalies in image data. More specifically, the abnormality degree of each image region is obtained by estimating its similarity to a dictionary of anomaly-free subregions in a training set.

- **Random Search [122, 93]**: Instead of learning a policy to optimize the search progress, random search generates a neural architecture randomly at each step. It has been widely demonstrated that random search is a strong baseline hard to be surpassed in NAS.

5.4.3 Experiment Setup

We train the child models on the training set under the anomaly-free settings, and update the controller on the validation set via the reward signal. The controller RNN is a two-layer LSTM with 50 hidden units on each layer. It is trained with the ADAM optimizer with a learning rate of $3.5e-4$. Weights are initialized uniformly in $[-0.1, 0.1]$. The search process is conducted for a total of 500 epochs. The size of the self-imitation buffer is 10. We use a Tanh constant of 2.5 and a

sample temperature of 5 to the hidden output of the RNN controller. We train the child models by utilizing a batch size of 64 and a momentum of 0.9 with the ADAM optimizer. The learning rate starts at 0.1, and is dropped by a factor of 10 at 50% and 75% of the training progress, respectively.

5.4.4 Evaluation Metrics

We adopt the following metrics to measure the effectiveness:

- **AUROC** [123] is the Area Under the Receiver Operating Characteristic curve, which is a threshold-independent metric [123]. The ROC curve depicts the relationship between TPR and FPR. The AUROC can be interpreted as the probability that a positive example is assigned a higher detection score than a negative example [124].

- **AUPR** [125] is the Area under the Precision-Recall curve, which is another threshold-independent metric [125, 126]. The PR curve is a graph showing the precision=TP/(TP+FP) and recall=TP/(TP+FN) against each other. The metrics **AUPR-In** and **AUPR-Out** denote the area under the precision-recall curve, where positive samples and negative samples are specified as positives, respectively.

- **RPRO** [116] stands for the relative per-region overlap. It denotes the pixel-wise overlap rate of the segmentations with the ground truth.

5.4.5 Results

5.4.5.1 Performance on Out-of-distribution Sample Detection

To answer the research question **Q1**, we compare `AutoOD` with the state-of-the-art handcrafted algorithms for the instance-level abnomral sample detection task using metrics AUROC, AUPR-In and AUPR-Out. Considering the automated search framework of `AutoOD`, we represent its performance by the best model found during the search process. In these experiments, we follow the setting in [117, 127]: Each model is trained on individual dataset $\mathcal{D}_{in}$, which is taken from MNIST, Fashion-MNIST, CIFAR-10, and Tiny-ImageNet, respectively. At test time, the

test images from $\mathcal{D}_{in}$ dataset can be viewed as the in-distribution (positive) samples. We sample out-of-distribution (negative) images from another real-world or synthetic noise dataset, after down-sampling/up-sampling and reshaping their sizes as the same as $\mathcal{D}_{in}$.

As can be seen from Table 5.3, in most of the test cases, the models discovered by `AutoOD` consistently outperform the handcrafted out-of-distribution detection methods with pre-trained models (ODIN [118]) and without pre-trained models (MSP [117]). It indicates that `AutoOD` could achieve higher performance in accuracy, precision, and recall simultaneously, with a more precise detection rate and fewer nuisance alarms.

5.4.5.2 Performance on Defect Sample Detection

To further answer question **Q1**, we test `AutoOD` on the pixel-level defect region segmentation task. The results from Table 5.2 show that `AutoOD` consistently outperforms the baseline methods by a large margin in terms of AUROC and RPRO. The higher AUROC demonstrates that the model found by `AutoOD` precisely detects images with defect sections out of the positive samples. The results also show that `AutoOD` has a better performance in RPRO. This indicates the search process of `AutoOD` helps the model to locate and represent the anomaly regions in negative images.

As a step-by-step concrete example, given input datasets from different benchmarks, we first define the global search space including definition hypothesis and distance measurement; and then define a 3-layers encoder-decoder in the local space, each layer contains output-channel, convolution kernel, pooling type, pooling kernel, normalization type and activation function. Table 5.4 illustrates the best architectures discovered by `AutoOD` on both instance-level abnormal sample detection and pixel-level defect region segmentation tasks.

5.4.5.3 Effectiveness of Curiosity-guided Search

To qualitatively evaluate the effectiveness of the curiosity-guided search for research question **Q2**, we perform ablation and hyperparameter analysis on Fashion-MNIST dataset with samples from CIFAR-10 as anomalies. Specifically, we control the hyperparameter η in Eq. (5.7) for illustration. Note that mathematical expression of $\eta = 0$ represents that there is no exploration.

From Table 5.5 (a) we can observe that: (1) The absence of exploration would negatively impact the final performance. The AUROC after 200 epochs could drop 1.9%. (2) The curiosity guided explorations help the controller find the optimal model faster. The better performance could be achieved in the 20-th, 100-th epochs when the controller has a larger weight η on explorations. This indicates that curiosity-guided search is a promising way for exploring more unseen spaces. (3) Comparing AUROC between $\eta = 0.01$ and $\eta = 0.1$, we observe that there is no significant increase in the final performance after 200 epochs. This indicates that a higher rate of explorations can not always guarantee a higher performance. (4) If we treat the performance of the searched result as Gaussian distributions, the standard deviations of the `AutoOD`'s performance keep increasing when η increases. This validates that the curiosity-guided search strategy increases the opportunity for the controller to generate child models in a more diverse way.

5.4.5.4 Effectiveness of Experience Replay

To further answer the question **Q2**, we evaluate the effectiveness of the experience replay buffers, by altering the size of the replay buffers $\mathcal{B}$ in Eq. (5.19). Corresponding results are reported in Table 5.5 (b). The results indicate that the increase of the buffer size could enhance model performance after 200 epochs. We also observe that the size of the buffer is sensitive to the final performance, as better performance would be achieved in the 20th, 100th epoch with a larger buffer size. This indicates that self-imitation learning based experience replay is useful in the search process. Larger buffer size brings benefits to exploit past good experiences.

5.4.5.5 Comparison Against Traditional NAS

Instead of applying the policy gradient based search strategy, one can use random search to find the best model. Although this baseline seems simple, it is often hard to surpass [122, 93]. We compare `AutoOD` with random search to answer the research question **Q3**. The quality of the search strategy can be quantified by the following three metrics: (1) the average performance of the top-5 models found so far, (2) the mean performance of the searched models in every 20 epochs, (3) the standard deviation of the model performance in every 20 epochs. From Fig. 5.3, we

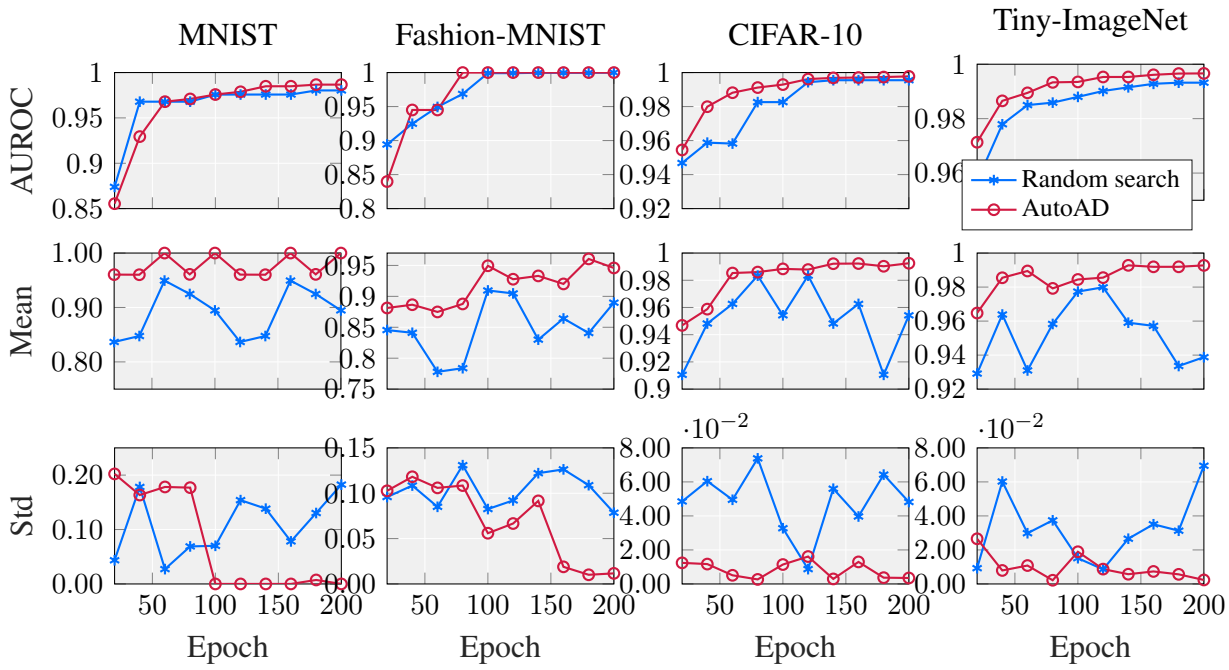


Figure 5.3: Performance comparison with random search.

can observe that: Firstly, our proposed search strategy is more efficient to find the well-performed models during the search process. As shown in the first row of Fig. 5.3, the performance of the top-5 models found by `AutoOD` consistently outperform the random search. The results also show that not only the best model of our search strategy is better than that of random search, but also the improvement of average top models is much more significant. This indicates that `AutoOD` explores better models faster than the random search. Secondly, there is a clear increasing tendency in the mean performance of `AutoOD`, which can not be observed in random search. It indicates that our search controller can gradually find better strategies from the past search experiences along the learning process, while the random search's controller has a relatively low chance to find a good child model. Thirdly, compared with the random search, there is a clear dropping of standard deviation along the search process. It verifies that our search strategy provides a more stable search process.

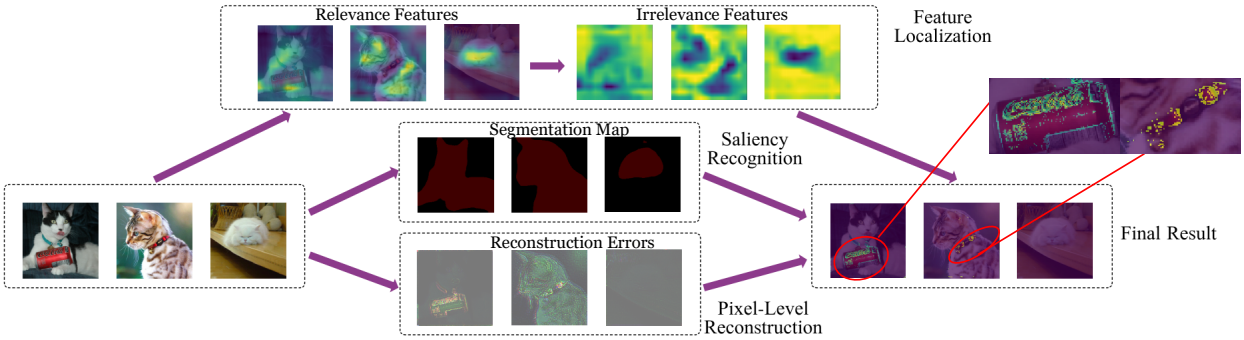


Figure 5.4: A case study of anomaly segmentation.

5.4.6 Case Study

To answer the research question **Q4**, we provide further analysis for the pixel-level defect region segmentation task, to get some insights about how to further improve the detection performance in more complicated real-world settings. To make the anomaly sections and the rest sections more distinguishable in the latent space, we use the reconstruction error to learn intrinsic representation for positive samples to extract common patterns. Yet, it is hard to directly apply `AutoOD` into more complicated, real-world settings. Due to the pure data-driven strategy, the reconstruction based denoters might be misled by background noises, other objects, or irrelevance features. We hereby introduce two strategies into `AutoOD` for regularization without increasing the model complexity.

Saliency Refinement via Target Object Recognition. We introduce a mask map s into the reconstruction based denoters from pre-trained models. It is used for localizing and identifying target salient object, in order to eliminate the negative effect caused by background noises and other objects in the same image. To concisely localize and identify the salient object, the key idea is to extract dense features for semantic segmentation. In our experiment, we introduce s from DeepLabV3 [128], which is pre-trained on PASCAL VOC 2012 [129].

Feature Augmentation via Gradient-based Localization. In order to amplify the contribution of the irrelevance features from the salient object, we introduce the feature augmentation map to re-weight the reconstruction result. We also introduce a coarse localization map to highlight

the irrelevance regions in the image from an interpretability perspective. Feature importance is reflected as gradients signal via backpropagation. The key idea is to use the gradient information flowing into the last convolutional layer of the CNN to assign importance values to each neuron for a particular decision of interest. Here, we follow the interpretation method from Grad-CAM [130, 131, 132, 133], which is designed to highlight important features, and pre-trained on VGG-16 [134]. The feature augmentation map is defined as opposite to the Grad-CAM: $\frac{1}{mn}\sum_m\sum_n\left(1-\frac{\partial y_i}{\partial A_{mn}}\right)$.

After the two steps above, we reweight the reconstructions:

$$\alpha_i = \|g\left(f\left(x_i;\mathcal{W}\right)\right)-x_i\|_2^2 \odot \frac{1}{mn}\sum_m\sum_n\left(1-\frac{\partial y_i}{\partial A_{mn}}\right)\odot \mathrm{s}_i \tag{5.20}$$

where $x_i \in \mathrm{R}^{m\times n}$ is a training sample and y_i is its target object class. $f(\cdot), g(\cdot)$ denotes encoder-decoder structures produced by `AutoOD`, and A denotes the feature map activation of a latent layer. We use a real-world dataset CAT [111] for illustration. We find the optimal model via `AutoOD` and get the pixel-level reconstruction map. Then, we further refine the map via saliency recognition and feature localization strategies. As can be seen from Fig. 5.4, `AutoOD` achieves better visualization results after applying the reweighting tricks discussed above. We also observe that the model can successfully identify the anomaly regions (cola bins, collars) within the salient objects (kitties). Meanwhile, it reduces the effect caused by the background noises and irrelevant objects.

(a) In-distribution dataset: MNIST

OOD Dataset	AUROC	AUPR In	AUPR Out
Fashion-MNIST	**99.9**/97.9/97.9	**99.9**/99.7/99.6	**100**/90.5/91.0
notMNIST	**99.8**/97.2/98.2	**99.8**/97.5/98.4	**100**/97.4/98.0
CIFAR-10	**99.9**/**99.9**/99.7	91.3/90.3/**99.9**	99.2/**99.9**/97.6
LSUN	**99.9**/**99.9**/99.8	99.9/96.8/**100**	**99.9**/99.2/99.0
Tiny-ImageNet	**99.9**/99.4/99.6	99.8/99.6/**99.9**	**99.8**/96.8/97.5
Gaussian	**99.9**/99.7/99.9	**100**/99.8/**100**	**100**/99.7/**100**
Uniform	**100**/99.9/**100**	**100**/99.9/**100**	**100**/99.9/**100**

(b) In-distribution dataset: Fashion-MNIST

OOD Dataset	AUROC	AUPR In	AUPR Out
MNIST	**99.9**/92.9/72.9	**99.9**/82.8/91.6	**99.9**/94.2/46.1
notMNIST	**99.8**/96.9/80.2	**99.1**/81.6/94.2	**100**/99.4/57.7
CIFAR-10	**99.9**/88.2/96.6	**99.5**/80.6/99.3	**99.9**/97.2/80.4
LSUN	**99.5**/89.7/96.0	97.9/81.9/**99.2**	**99.9**/97.7/79.9
Tiny-ImageNet	**98.2**/87.7/95.5	90.7/80.4/**99.0**	**95.3**/97.1/82.5
Gaussian	**99.9**/97.2/89.6	**99.9**/82.24/98.0	**100**/99.5/48.2
Uniform	**99.9**/95.8/63.6	**99.9**/82.9/91.4	**99.9**/99.0/19.8

(c) In-distribution dataset: CIFAR-10

OOD Dataset	AUROC	AUPR In	AUPR Out
MNIST	**100**/98.4/99.9	**100**/99.4/**100**	**100**/89.4/99.4
Fashion-MNIST	**99.6**/98.2/99.4	98.1/96.1/**99.9**	**99.9**/98.8/97.3
notMNIST	**99.9**/96.8/98.1	**99.4**/99.0/99.2	**100**/89.4/90.2
LSUN	80.0/75.2/**85.6**	81.2/73.1/**83.5**	**85.8**/73.3/85.1
Tiny-ImageNet	**84.0**/72.6/81.6	**87.5**/73.5/76.9	**87.8**/80.6/84.8
Gaussian	**99.9**/86.3/98.8	**99.9**/90.5/99.1	**99.3**/77.0/97.9
Uniform	**99.9**/86.4/99.0	**99.9**/90.2/99.2	**99.9**/78.6/98.6

(d) In-distribution dataset: Tiny-ImageNet

OOD Dataset	AUROC	AUPR In	AUPR Out
MNIST	**100**/99.8/94.8	**100**/98.2/98.9	**100**/98.2/79.9
Fashion-MNIST	**99.7**/70.4/73.8	**98.6**/88.4/92.6	**100**/85.5/39.5
notMNIST	**99.9**/80.6/82.7	**99.7**/90.4/95.2	**100**/85.8/56.0
CIFAR-10	**86.7**/82.9/58.0	**95.8**/75.3/85.7	**89.1**/75.3/28.2
LSUN	**88.6**/74.2/55.6	92.7/**99.5**/86.7	93.9/**99.5**/18.7
Gaussian	**99.9**/97.0/95.4	**100**/98.0/99.0	**99.9**/94.8/80.5
Uniform	**100**/96.0/87.5	**100**/97.4/96.8	**100**/99.3/62.8

Table 5.3: Performance comparison on instance-level abnormal sample detection.

Actions	MNIST	Fashion-MNIST	CIFAR-10	Tiny-ImageNet	MVTec-AD
Definition hypothesis	reconstruction	cluster	centroid	reconstruction	reconstruction
Distance measurement	l_1	$l_{2,1}$	$l_{2,1}$	$l_{2,1}$	$l_{2,1}$+ SSIM
Layer-1					
Output-channel	32	16	16	16	32
Convolution kernel	5×5	1×1	1×1	5×5	1×1
Pooling type	mean	average	average	average	average
Pooling kernel	1×1	1×1	1×1	1×1	3×3
Normalization type	no	no	no	no	no
Activation function	ReLU	Sigmoid	Sigmoid	Linear	LeakyReLU
Layer-2					
Output-channel	8	32	32	32	32
Convolution kernel	3×3	5×5	5×5	3×3	5×5
Pooling type	average	mean	mean	mean	mean
Pooling kernel	1×1	7×7	7×7	7×7	5×5
Normalization type	no	no	no	no	batch
Activation function	ELU	ReLU6	Sigmoid	Softplus	Tanh
Layer-3					
Output-channel	8	16	16	16	16
Convolution kernel	7×7	1×1	1×1	7×7	1×1
Pooling type	average	average	average	average	mean
Pooling kernel	5×5	1×1	1×1	1×1	5×5
Normalization type	no	instance	instance	instance	instance
Activation function	ReLU6	LeakyReLU	Sigmoid	LeakyReLU	Tanh

Table 5.4: The architectures discovered by `AutoOD` for MNIST, Fashion-MNIST, CIFAR-10, Tiny-ImageNet, and MVTec-AD.

(a) Curiosity-guided Search

	AUROC_{20}	AUROC_{100}	AUROC_{200}	mean_{200}	std_{200}
η=0	85.27	96.23	96.51	90.01	0.093
η=0.01	85.46	96.54	98.50	91.20	0.097
η=0.1	85.46	96.93	98.41	91.84	0.131

(b) Experience Replay Buffer

	AUROC_{20}	AUROC_{100}	AUROC_{200}
no buffer	85.43	96.57	98.00
buffer size=5	88.05	97.12	98.04
buffer size=10	87.44	97.70	98.12

Table 5.5: Ablations and parameter analysis on Fashion-MNIST (In-distribution: normal data) and CIFAR-10 (Out-of-distribution: anomalies).

6. AN END-TO-END AUTOMATED ANOMALY DETECTION SYSTEM[1]

In this chapter, we aim to propose an end-to-end outlier detection system. Current outlier detection techniques are often manually designed for specific domains, requiring large human efforts of database setup, algorithm selection, and hyper-parameter tuning. To fill this gap, the outcome algorithm is expected as an automated end-to-end Python system for Outlier Detection with Database Support(`PyODDS`), which automatically optimizes an outlier detection pipeline for a new data source at hand.

6.1 Motivation

Outliers refer to the objects with patterns or behaviors that are significantly rare and different from the rest of the majority. Outlier detection plays an important role in various applications, such as fraud detection, cyber security, medical diagnosis, and industrial manufacturer. The research of outlier detection traces far back, and numerous approaches have been proposed to tackle the problem. Representative categories of outlier detection approaches include density-based, distance-based and model-based approaches.

Despite the exciting results in outlier detection research, it is challenging and expensive to apply outlier detection to tackle real-world problems. First, there is no single outlier detection algorithm outperforms the others on all scenarios, since many outlier detection techniques have been specifically developed for certain application domains [135, 136, 137]; Second, most outlier detection methods highly depend on their hyper-parameter settings; Third, the contamination ratio of outliers in the given task is usually unknown.

Recently, efforts have been made to integrate various outlier detection algorithms into a single package. Existing approaches [138, 139] contain different outlier detection methods with various programming languages, yet they do not tackle with optimal pipeline design as searching and

[1]This chapter is reprinted with permission from "Pyodds: An end-to-end outlier detection system with automated machine learning", by Yuening Li, Daochen Zha, Praveen Venugopal, Na Zou, Xia Hu, 2020. Companion Proceedings of the Web Conference 2020. Copyright 2020 by IW3C2.

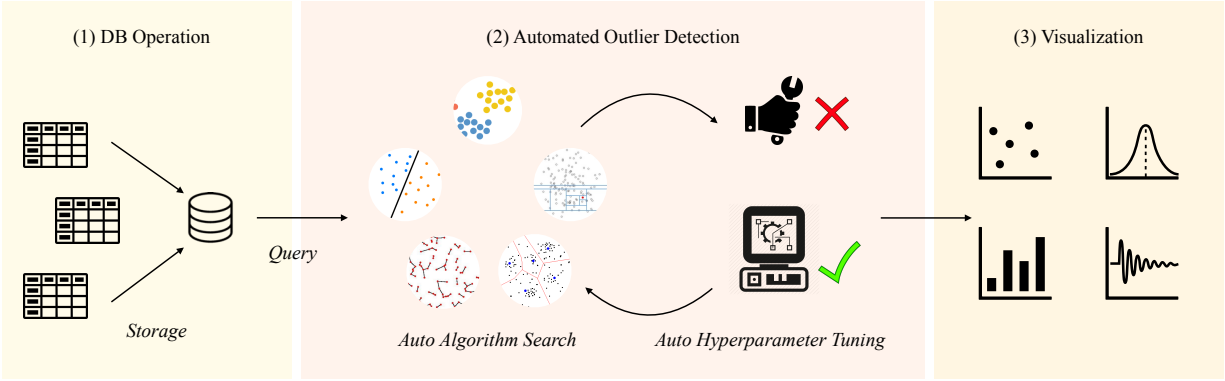


Figure 6.1: Overview of PyODDS

exploration problems, and do not cater specifically to backend-servers for large-scale applications.

In the meanwhile, a large focus of the machine learning community has been to find better hyper-parameter settings, which has been successfully tackled using Bayesian optimization, reinforcement learning, etc., and forms a core component of AutoML systems. However, less attention has been paid to finding a good solution for an end-to-end, joint optimization problem including multiple components, especially in real-world data mining tasks.

To bridge the gap, we present PyODDS, a full-stack, end-to-end system for outlier detection. PyODDS has desirable features from the following perspectives. First, to our best knowledge, PyODDS describes the first attempt to incorporate automated machine learning with outlier detection, and belongs to one of the first attempts to extend automated machine learning concepts into real-world data mining tasks. Second, we carefully design an end-to-end framework for outlier detection, including database operations and maintenance, the search process of automated outlier detection (including the search space and the search strategy design). Finally, we present a visual analytic system based on our proposed framework for demonstration.

6.2 PyODDS System Framework

The pipeline from query data to evaluation and visualization is outlined in Fig. 6.1, which consists of 3 components. The first component is the information extraction, which collects the source data via query functions with flexible time-slices segmentation, including user-info confirmation,

database operation, and maintenance. The second component is the suspicious outlier detection. It detects suspicious instances with traditional outlier detection approaches as an automated machine learning problem, including the search space design and the search strategy development. The last component is the visualization part, which designs for users to understand the detection results better. In the following subsections, we focus on the first and second components.

6.2.1 Information Extraction

In this component, we extract the information from a specific time range through database operations. `PyODDS` includes database operation functions for client users: (1) $connect\_server$ function allows the client to connect the server with host address and user information for safety verification; (2) $query\_data$ function designs for flexible time-slices segmentation.

6.2.2 Automated Outlier Detection

To detect suspicious outliers, we need to find the best pipeline configuration. We formulate the problem of finding the best policy as a conjunctive search problem. In this component, our method consists of two subsections: a search space and a search strategy.

6.2.2.1 Search Space

In our search space, a policy consists of sub-policies as a batch of outlier detection algorithms. Additionally, the policy also contains the hyper-parameters as another conditional sub-policy: 1) specific hyper-parameter settings corresponding to each algorithm sub-policy which controls the learning process; 2) the contamination ratio which determines the portion of outliers corresponding to the given data source.

Each algorithm we included also comes with a default range of hyper-parameter settings. Within each algorithm sub-policy, hyper-parameters which might be discrete, ordinal, or continuous, need to be optimized in the meanwhile.

6.2.2.2 *Search Strategy*

Following the search space setting we proposed above, we define the problem of automated outlier detection with algorithm selection and hyper-parameter tuning as follows.

Let $\mathcal{A} = \{A_1, A_2, ..., A_N\}$ be a set of outlier detection algorithms, and $\boldsymbol{\lambda} = \{\lambda_1, \lambda_2, ..., \lambda_N\}$ be the set of corresponding hyper-parameters. We assume $\boldsymbol{\lambda}$ is given. Let $\mathcal{D}^{train}$ and $\mathcal{D}^{val}$ be the training set and validation set, respectively. Denote $\mathcal{M}(A^s, \boldsymbol{\lambda}^s, \mathcal{D}^{train}, \mathcal{D}^{val})$ as the performance on $\mathcal{D}^{val}$ in terms of metric M when trained on D^{train} with algorithm $A^s \subseteq \mathcal{A}$ and corresponding hyper-parameters $\boldsymbol{\lambda}^s \subseteq \boldsymbol{\lambda}$. The algorithm is to find optimal solution $A^*, \boldsymbol{\lambda}^*$ via observation history $\mathcal{H}$. We define the objective as

$$A^*, \boldsymbol{\lambda}^* \in \underset{A^s \subseteq \mathcal{A}, \boldsymbol{\lambda}^s \subseteq \boldsymbol{\lambda}^*}{\operatorname{argmax}} M(A^s, \boldsymbol{\lambda}^s, \mathcal{D}^{train}, \mathcal{D}^{val}). \tag{6.1}$$

To get a step further, Sequential Model-Based Global Optimization (SMBO) algorithms have been used in many applications where evaluation of the fitness function is expensive, i.e., automated machine learning tasks [140]. To optimize the evaluation function $\mathcal{M}(\mathcal{A}^s, \boldsymbol{\lambda}^s, \mathcal{D}^{train}, \mathcal{D}^{val})$, we optimize the criterion of Expected Improvement, the expectation under $\mathcal{A}^s, \boldsymbol{\lambda}^s$ when $y = \mathcal{M}(\mathcal{A}^s, \boldsymbol{\lambda}^s, \mathcal{D}^{train}, \mathcal{D}^{val})$ negatively exceed the threshold y^*:

$$\mathrm{EI}_{y^*}(x) = \int_{-\infty}^{y^*} (y^* - y)\, p(y|x) dy = \int_{-\infty}^{y^*} (y^* - y)\, \frac{p(x|y)p(y)}{p(x)} dy, \tag{6.2}$$

where the point x^* that maximizes the surrogate (or its transformation) becomes the proposal for where the function should be evaluated.

The tree-structured Parzen estimator (TPE) models $p(x|y)$ by transforming to a generative process, which replaces the distributions of the configuration prior to non-parametric densities. We borrow the strategy in [140] here to minimize the EI. We keep the Estimation of Distribution (EDA, [141]) approach on the discrete part of our search space (algorithm selection and discrete hyper-parameters), where we sample candidate points according to binomial distributions, while we use

Algorithm 4 Optimization Process

Input: $\mathcal{H}, \mathcal{A}, \boldsymbol{\lambda}, \mathcal{D}^{train}, \mathcal{D}^{val}, T_{max}$
$T \leftarrow 1$,
while $T < T_{max}$ **do**
    $T \leftarrow T + 1$
    $A^t \leftarrow M(\mathcal{A}^{t-1}, \boldsymbol{\lambda}^{t-1}, \mathcal{D}^{train}, \mathcal{D}^{val})$
    $\lambda^t_{\text{discrete}} \leftarrow M(\mathcal{A}^t, \boldsymbol{\lambda}^{t-1}, \mathcal{D}^{train}, \mathcal{D}^{val})$
    $\lambda^t_{\text{continuous}} \leftarrow M(\mathcal{A}^t, \boldsymbol{\lambda}^{t-1}, \mathcal{D}^{train}, \mathcal{D}^{val})$
    $\mathcal{H} \leftarrow \mathcal{H} \cup \{\mathcal{A}^t, \lambda^t\}$
end while
Return A^*, λ^* with the best performance in $\mathcal{H}$

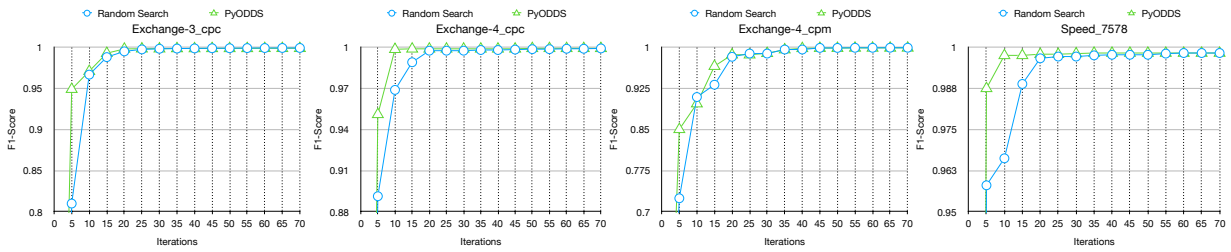


Figure 6.2: Progression of top-5 averaged performance of different search methods, i.e., Random Search and `PyODDS`.

the Covariance Matrix Adaptation - Evolution Strategy, a gradient-free evolutionary algorithm (CMA-ES, [142]) for the remaining part of our search space (continuous hyper-parameters). The whole optimization process can be summarized in Algorithm 4.

6.3 Experimental Evaluation

6.3.1 Data Source

The time-series data, which is used to train and evaluate `PyODDS` , comes from a benchmark dataset, NAB corpus [143]. NAB corpus contains 58 different individual tasks with ground-truth. The reasons why we employ this data source are in three folds. First, NAB corpus provides fine-grained labels, where the core principles in independence, transparency and fairness guarantee. Second, the data in NAB corpus are ordered, timestamped, which cover a varies of real-world ap-

plication scenarios, including server monitor logs from AmazonCloudwatch service, online advertisement clicking-rates, real-time traffic transportation, and collection from Twitters with trading related contents. Third, each raw data file is a dictionary of key-value pair, which is naturally to be represented as tabular data that meets the requirements of the backend database service in the `PyODDS`.

6.3.2 Algorithm Space Configurations

We implemented 13 state-of-the-arts outlier detection algorithms as the search space, including statistical approaches, and recent neural network frameworks. In the meanwhile, in order to support both static and time series data analysis, the search space covers algorithms with different settings.

6.3.3 Detection Results Evaluation

In this section, we empirically investigate the performance of `PyODDS` to answer the following questions: first, how does the algorithm with hyper-parameters discovered by `PyODDS` compare with state-of-the-art handcrafted algorithms? Second, how does the search process affect performance?

In Table 6.1, we show the performance on the NAB corpus. We follow the default setting in NAB as the scoring algorithm, which uses a scaled sigmoidal scoring function to quantify the detection performance. The smooth score function ensures that small labeling errors will not cause large changes in reported scores. The evaluation matrix includes the standard profile, reward low FPs, and reward low FNs. The standard profile assigns TPs, FPs, and FNs with relative weights, and the latter two profiles accredit greater penalties for FPs and FNs, respectively. For more detailed definitions, please refer to the default setting [144].

To answer the first question, we use `PyODDS` to find the best policies on the NAB corpus. As can be seen from Table 6.1, the outlier detection solution discovered by `PyODDS` architecture achieves competitive performance with current state-of-the-art models: the handcrafted algorithms, and random searched results. It shows that `PyODDS` could find optimal solutions within a large range of configurations for different detection tasks.

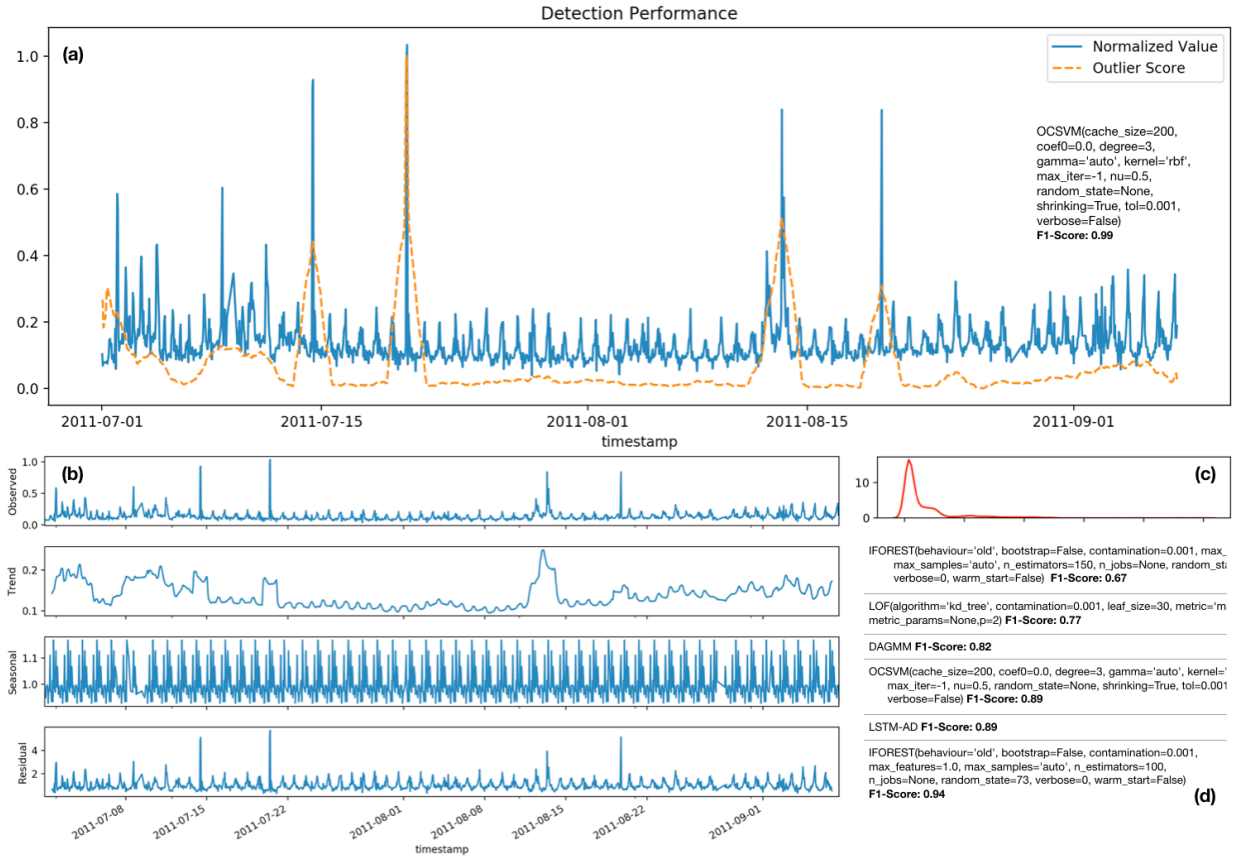


Figure 6.3: Demonstration of using `PyODDS` in visualizing prediction result

Model	Standard Profile	Reward Low FP	Reward Low FN
Perfect	100	100	100
CBLOF [145]	94.56	93.29	96.68
HBOS [146]	91.86	95.74	93.47
IFOREST [147]	92.44	92.10	94.38
KNN [148]	90.76	96.12	93.42
LOF [52]	92.61	88.78	89.86
OCSVM [149]	88.63	94.60	91.31
PCA [150]	93.15	94.50	96.28
RobustCovariance	96.68	95.27	94.76
SOD [151]	78.46	78.46	82.93
AUTOENCODER [15]	94.74	96.41	93.64
DAGMM [16]	85.27	83.35	90.21
LSTMAD [152]	93.19	95.18	92.43
LSTMENCDOC [153]	94.31	89.23	89.23
RANDOM	87.38	90.79	86.90
PyODDS	96.68	95.27	94.76

Table 6.1: Performance comparison for outlier detection algorithms.

For the second question, we conduct the search process in the same search space with different search strategies. As can be seen from Fig. 6.2, PyODDS is more efficient in finding the well-performed architectures during the search progress. Comparing with the random search, the top-5 architectures discovered by PyODDS have better performance (F1-score) and could convergence faster on different datasets. It shows the effectiveness of the search strategy PyODDS implemented could enhance the performance and accelerate the search efficiency.

6.4 Demonstration

PyODDS is composed of a frontend and a server backend. Our system is written in Python and uses Apache Spark as the server backend and TDengine as the database support service. We demonstrate our system based on the real-world datasets from the Numenta Anomaly Benchmark (NAB) corpus [143].

First, after selecting the data source and time range, our system will automatically find an algorithm with default hyper-parameter settings from the search space, and show the detection

results. Illustrated by Fig. 6.3(a), we provide the normalized value from the original time series as the blue line, and outlier score as orange line, to help users understand the data distribution in the original data source, as well as the detection results. Lower outlier score indicates that the data point is considered "normal". Higher values indicate the presence of an outlier in the data.

In addition, `PyODDS` provides time series analysis tools for users to better understand the data source. Illustrated in Fig. 6.3(b), `PyODDS` decomposes the original time series as a combination of level, trend, seasonality, and residual components. The residual values could also act as denoters for outlier detection in time-series. In the meanwhile, in Fig. 6.3(c), `PyODDS` estimates the probability density function of the values in each timestamp, which provides a comprehensive scope of the data distribution in the original data source. According to the search strategy and search space we proposed in the previous sections, we also provide trace logs to illustrate the search process for records. After several iterations of the search process, the selected algorithms with specific configurations are listed in Fig. 6.3(d)). As shown in the user case, extreme values and spikes without seasonal patterns (i.e., in the time stamp 2011-07-15, etc) have larger outlier score than the rest majority as normal cases (shown in (a) and (c)), as well as larger residual value after time series decomposition (shown in (b)). Current best solution is the sub-policy OCSVM with specific hyperparameter settings.

7. CONCLUSIONS AND FUTURE RESEARCH OPPORTUNITIES

In this dissertation, we propose a series of methods and frameworks to address different problems in the process of anomaly detection in complex data structures. In this chapter, we conclude the dissertation and propose the problems to be studied in the future following this dissertation.

7.1 Conclusions

First, we introduce an effective framework SpecAE for identifying anomalies in attributed networks. To detect global and community anomalies, we map the attributed network into two types of low-dimensional representations. The first type consists of node representations learned from an autoencoder and corresponding reconstruction errors. To learn the second type of representations, we design the novel graph deconvolution neural networks as the complementary operation to the graph convolution, aiming to reconstruct nodal attributes according to the topological relations. The two types of learned representations are applied to a Gaussian mixture model to perform anomaly detection. The tailored representation learning model and the GMM model are trained collaboratively.

Second, we propose an novel cross-modal anomaly detection approach CMAD based on deep neural networks. The proposed CMAD framework is able to identify inconsistent patterns or behaviors of instances across different modalities by capturing their nonlinear correlations in the learned consensus latent feature space. Firstly, we train deep structured model to represent features from different modalities, and then project the features into a consensus latent feature space. Secondly, we "pull" the projections of a pair of instances from different modalities together if their cross-modal patterns are consistent, while "push" them further apart otherwise. Finally, we distinguish the cross-modality anomalies by measuring the distances across different modalities.

Third, we investigate a novel and challenging problem of learning disentangled time-series representations. DTS introduces a multi-level disentanglement strategy, covering both individual latent factor and group semantic segments, to generate hierarchical semantic concepts as the inter-

pretable and disentangled representation. It alleviates the KL vanishing problem via introducing a mutual information maximization term, while preserving a heavier penalty on the total correlation and the dimension-wise KL to keep the disentangle property via balancing the preference between correct inference and fitting data distribution.

Fourth, we investigated a novel and challenging problem of automated deep model search for anomaly detection. Different from the existing NAS methods that focus on discovering effective deep architectures for supervised learning tasks, we proposed AutoOD, an automated unsupervised anomaly detection framework, which aims to find an optimal neural network model within a predefined search space for a given dataset. AutoOD builds on the theory of curiosity-driven exploration and self-imitation learning. It overcomes the curse of local optimality, the unfair bias, and inefficient sample exploitation problems in the traditional search methods.

Fifth, we propose an end-to-end approach to detect outliers, and demonstrate the prediction results for users to better understand the data source. PyODDS automatically search an optimal outlier detection pipeline for a new dataset at hand out of a defined proposed search space via the proposed search strategy.

With these methods and frameworks, we successfully address the challenges in the process of anomaly detection with complex data structures. It covers modeling complex data structures for anomaly detection settings, including multimodal structures, topological relation and sequential correlations. It also discusses how to automatically search an optimal anomaly detection framework for a given dataset with specific data characteristics.

7.2 Future Work

In the future, the following open questions may be studied following this dissertation.

- **Distance measures for anomaly detection.** Distance measurement stands for the matrix measuring the distance between observations in the latent spaces for the anomaly detection purpose. Traditional measurement ways include Euclidean distance, Cosine distance, l_1, l_2, $l_{2,1}$ norms. But different scenarios bring new challenges to directly port traditional

measurement functions. For instance, MST-based distances [154] approximate the geodesic distances among vertices, which works better than the Euclidean distance in the presence of nonlinear manifold structure. SSIM [155] is used for measuring the similarity between two images. The anomaly degree for images could contain additional structural information while also incorporating important perceptual phenomena. To further invest various distance measurements is important for different data structures.

- **Enhance interpretability in anomaly detection.** With more and more advanced anomaly detection models being proposed, the interpretation of the detection model, especially deep structured models, becomes more and more difficult. Despite the successes, machine learning has its own limitations and drawbacks. The most significant one is the lack of transparency behind their behaviors, which leaves users with little understanding of how particular decisions are made by these models. For instance, an advanced autonomous vehicles, the interpretability of uncertainty and unusual object detection could be utilized to guild the perception, localization, motion planning and controlling systems.

- **Model other complex data structures.** In many real-world scenarios, the data we collected often comes from different sources or can be represented by different feature formats. Traditional methods lack the capability to model these complex data structures for anomaly detection. For instance, in the financial domain, semi-structured or unstructured text corpora play importance role for anti-fraud, cyber security and automatic transactions, including financial reports, press releases, earnings call transcripts, credit agreements, news articles, customer interaction logs and social data.

- **Create anomaly detection benchmarks for automated model design and tuning.** Though we have many anomaly detection datasets and models, there still lacks a comprehensive benchmark for anomaly detection model design and tuning, especially for large-scale and heterogeneous datasets.

REFERENCES

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[2] J. R. Dorronsoro, F. Ginel, C. Sgnchez, and C. Cruz, "Neural fraud detection in credit card operations," *IEEE transactions on neural networks*, vol. 8, no. 4, pp. 827–834, 1997.

[3] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[4] A. Pantelopoulos and N. G. Bourbakis, "A survey on wearable sensor-based systems for health monitoring and prognosis," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 1, pp. 1–12, 2010.

[5] C.-I. Chang and S.-S. Chiang, "Anomaly detection and classification for hyperspectral imagery," *IEEE transactions on geoscience and remote sensing*, vol. 40, no. 6, pp. 1314–1325, 2002.

[6] F. Gonzalez, D. Dasgupta, and R. Kozma, "Combining negative selection and classification techniques for anomaly detection," in *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, vol. 1, pp. 705–710, IEEE, 2002.

[7] L. Portnoy, E. Eskin, and S. Stolfo, "Intrusion detection with unlabeled data using clustering," in *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001*, Citeseer, 2001.

[8] L. Ruff, N. Görnitz, L. Deecke, S. A. Siddiqui, R. Vandermeulen, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *International Conference on Machine Learning*, 2018.

[9] D. Zhou, J. He, K. S. Candan, and H. Davulcu, "Muvir: Multi-view rare category detection.," in *IJCAI*, pp. 4098–4104, 2015.

[10] X. Huang, J. Li, and X. Hu, "Label informed attributed network embedding," in *WSDM*, 2017.

[11] N. Liu, X. Huang, and X. Hu, "Accelerated local anomaly detection via resolving attributed networks," in *IJCAI*, 2017.

[12] J. Gao, W. Fan, D. Turaga, S. Parthasarathy, and J. Han, "A spectral framework for detecting inconsistency across multi-source object relationships," in *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pp. 1050–1055, IEEE, 2011.

[13] J. Li, H. Dani, X. Hu, and H. Liu, "Radar: Residual analysis for anomaly detection in attributed networks," *IJCAI*, 2017.

[14] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *NIPS*, 2016.

[15] S. Hawkins, H. He, G. Williams, and R. Baxter, "Outlier detection using replicator neural networks," in *International Conference on Data Warehousing and Knowledge Discovery*, pp. 170–180, Springer, 2002.

[16] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," *ICLR*, 2018.

[17] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *NIPS*, 2017.

[18] B. Thompson, "Canonical correlation analysis.," 2000.

[19] P. L. Lai and C. Fyfe, "Kernel and nonlinear canonical correlation analysis," *International Journal of Neural Systems*, vol. 10, no. 05, pp. 365–377, 2000.

[20] L. Sun, S. Ji, and J. Ye, "Canonical correlation analysis for multilabel classification: A least-squares formulation, extensions, and analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 194–200, 2011.

[21] K. Chaudhuri, S. M. Kakade, K. Livescu, and K. Sridharan, "Multi-view clustering via canonical correlation analysis," in *Proceedings of the 26th annual international conference on machine learning*, pp. 129–136, ACM, 2009.

[22] M. B. Blaschko and C. H. Lampert, "Correlational spectral clustering," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, IEEE, 2008.

[23] S. Li, M. Shao, and Y. Fu, "Multi-view low-rank analysis for outlier detection," in *Proceedings of the 2015 SIAM International Conference on Data Mining*, pp. 748–756, SIAM, 2015.

[24] L. Zhang, S. Wang, X. Zhang, Y. Wang, B. Li, D. Shen, and S. Ji, "Collaborative multi-view denoising," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2045–2054, ACM, 2016.

[25] T. Mikolov, Q. V. Le, and I. Sutskever, "Exploiting similarities among languages for machine translation," *arXiv preprint arXiv:1309.4168*, 2013.

[26] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 689–696, 2011.

[27] R. Salakhutdinov and H. Larochelle, "Efficient learning of deep boltzmann machines," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 693–700, 2010.

[28] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pp. 6645–6649, IEEE, 2013.

[29] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[30] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010.

[31] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view rgb-d object dataset," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1817–1824, IEEE, 2011.

[32] S. Akaho, "A kernel method for canonical correlation analysis," *arXiv preprint cs/0609071*, 2006.

[33] H. Abdi, "Partial least square regression (pls regression)," *Encyclopedia for research methods for the social sciences*, vol. 6, no. 4, pp. 792–795, 2003.

[34] L. Wang, Y. Li, J. Huang, and S. Lazebnik, "Learning two-branch neural networks for image-text matching tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[35] L. Wang, Y. Li, and S. Lazebnik, "Learning deep structure-preserving image-text embeddings," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5005–5013, 2016.

[36] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang, "Deep structured energy based models for anomaly detection," *arXiv preprint arXiv:1605.07717*, 2016.

[37] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 119–128, ACM, 2015.

[38] J. Liu, E. Bier, A. Wilson, J. A. Guerra-Gomez, T. Honda, K. Sricharan, L. Gilpin, and D. Davies, "Graph analysis for detecting fraud, waste, and abuse in healthcare data," *AI Magazine*, 2016.

[39] V. Van Vlasselaer, C. Bravo, O. Caelen, T. Eliassi-Rad, L. Akoglu, M. Snoeck, and B. Baesens, "Apate: A novel approach for automated credit card transaction fraud detection using network-based extensions," *Decision Support Systems*, 2015.

[40] W. Song, H. Yin, C. Liu, and D. Song, "Deepmem: Learning graph neural network models for fast and robust memory forensic analysis," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 606–618, ACM, 2018.

[41] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual review of sociology*, vol. 27, no. 1, pp. 415–444, 2001.

[42] W. Yu, W. Cheng, C. C. Aggarwal, K. Zhang, H. Chen, and W. Wang, "Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks," in *SIGKDD*, 2018.

[43] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *ICLR*, 2017.

[44] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *AAAI*, 2018.

[45] J. Chen, S. Sathe, C. Aggarwal, and D. Turaga, "Outlier detection with autoencoder ensembles," in *Proceedings of the 2017 SIAM International Conference on Data Mining*, pp. 90–98, SIAM, 2017.

[46] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, p. 4, ACM, 2014.

[47] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *Special Lecture on IE*, vol. 2, pp. 1–18, 2015.

[48] T. Ma, L. Li, S. Ji, X. Wang, Y. Tian, A. Al-Dhelaan, and M. Al-Rodhaan, "Optimized laplacian image sharpening algorithm based on graphic processing unit," *Physica A: Statistical Mechanics and its Applications*, vol. 416, pp. 400–410, 2014.

[49] B. Perozzi, L. Akoglu, P. Iglesias Sánchez, and E. Müller, "Focused clustering and outlier detection in large attributed graphs," in *SIGKDD*, 2014.

[50] D. B. Skillicorn, "Detecting anomalies in graphs," in *Intelligence and Security Informatics, 2007 IEEE*, 2007.

[51] X. Song, M. Wu, C. Jermaine, and S. Ranka, "Conditional anomaly detection," *TKDE*, 2007.

[52] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *ACM sigmod record*, 2000.

[53] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Advances in neural information processing systems*, 2000.

[54] Y. Shen, C. Yang, X. Tang, and B. Zhou, "Interfacegan: Interpreting the disentangled face representation learned by gans," *TPAMI*, 2020.

[55] Z. Jiang and et al, "Variational deep embedding: An unsupervised and generative approach to clustering,"

[56] R.-Q. Chen and et al, "A joint model for anomaly detection and trend prediction on it operation series," *arXiv preprint arXiv:1910.03818*, 2019.

[57] S. Semeniuta and et al, "A hybrid convolutional variational autoencoder for text generation," *arXiv preprint arXiv:1702.02390*, 2017.

[58] V. Fortuin and et al, "Som-vae: Interpretable discrete representation learning on time series," *arXiv preprint arXiv:1806.02199*, 2018.

[59] V. Fortuin, D. Baranchuk, G. Rätsch, and S. Mandt, "Gp-vae: Deep probabilistic time series imputation," in *AISTAT*, 2020.

[60] X. Guo, L. Zhao, Z. Qin, L. Wu, A. Shehu, and Y. Ye, "Interpretable deep graph generation with node-edge co-disentanglement," in *KDD*, 2020.

[61] H. Shao, S. Yao, D. Sun, A. Zhang, S. Liu, D. Liu, J. Wang, and T. Abdelzaher, "Controlvae: Controllable variational autoencoder," in *ICML*, 2020.

[62] R. T. Chen, X. Li, R. Grosse, and D. Duvenaud, "Isolating sources of disentanglement in variational autoencoders," *arXiv preprint arXiv:1802.04942*, 2018.

[63] P. Z. Wang and W. Y. Wang, "Riemannian normalizing flow on variational wasserstein autoencoder for text modeling," *arXiv preprint arXiv:1904.02399*, 2019.

[64] X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel, "Variational lossy autoencoder," *arXiv:1611.02731*, 2016.

[65] J. Chung, K. Kastner, L. Dinh, K. Goel, A. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," *arXiv preprint arXiv:1506.02216*, 2015.

[66] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[67] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, *et al.*, "Understanding disentangling in beta-vae," *arXiv preprint arXiv:1804.03599*, 2018.

[68] S. Watanabe, "Information theoretical analysis of multivariate correlation," *IBM Journal of research and development*, 1960.

[69] S. Zhao, J. Song, and S. Ermon, "Infovae: Balancing learning and inference in variational autoencoders," in *AAAI*, 2019.

[70] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length, and helmholtz free energy," *NeurIPS*, 1994.

[71] S. Bai *et al.*, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[72] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *ICML*, 2015.

[73] R. Cai, Z. Li, P. Wei, J. Qiao, K. Zhang, and Z. Hao, "Learning disentangled semantic representation for domain adaptation," in *IJCAI*, 2019.

[74] N. Tishby and N. Zaslavsky, "Deep learning and the information bottleneck principle," in *2015 IEEE Information Theory Workshop (ITW)*, 2015.

[75] E. Tzeng and et al, "Deep domain confusion: Maximizing for domain invariance," *arXiv preprint arXiv:1412.3474*, 2014.

[76] G. Wilson, J. R. Doppa, and D. J. Cook, "Multi-source deep domain adaptation with weak supervision for time-series sensor data," in *KDD*, 2020.

[77] F. Locatello and et al, "Challenging common assumptions in the unsupervised learning of disentangled representations," in *ICML*, 2019.

[78] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones.," in *Esann*, 2013.

[79] A. Stisen and et al, "Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition," in *SenSys*, 2015.

[80] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SigKDD Explorations Newsletter*, 2011.

[81] J. Liu and et al, "uwave: Accelerometer-based personalized gesture recognition and its applications," *Pervasive and Mobile Computing*, 2009.

[82] H. Blackburn and et al, "The electrocardiogram in population studies: a classification system,"

[83] H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, and M. Marchand, "Domain-adversarial neural networks," *arXiv preprint arXiv:1412.4446*, 2014.

[84] S. Purushotham, W. Carvalho, T. Nilanon, and Y. Liu, "Variational recurrent adversarial deep domain adaptation.," in *ICLR*, 2017.

[85] S. Ben-David and R. Urner, "Domain adaptation–can quantity compensate for quality?," *Annals of Mathematics and Artificial Intelligence*, 2014.

[86] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," in *ICML*, 2018.

[87] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *ICLR*, 2016.

[88] G. Swirszcz, W. M. Czarnecki, and R. Pascanu, "Local minima in training of neural networks," *arXiv preprint arXiv:1611.06310*, 2016.

[89] X. Chu, B. Zhang, R. Xu, and J. Li, "Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search," *arXiv preprint arXiv:1907.01845*, 2019.

[90] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *KDD*, pp. 665–674, 2017.

[91] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *JMLR*, 2019.

[92] Y. Li, D. Zha, P. Venugopal, N. Zou, and X. Hu, "Pyodds: An end-to-end outlier detection system with automated machine learning," in *Companion Proceedings of the Web Conference 2020*, 2020.

[93] L. Li and A. Talwalkar, "Random search and reproducibility for neural architecture search," *arXiv preprint arXiv:1902.07638*, 2019.

[94] C. Liu, L.-C. Chen, F. Schroff, H. Adam, W. Hua, A. L. Yuille, and L. Fei-Fei, "Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation," in *CVPR*, pp. 82–92, 2019.

[95] X. Gong, S. Chang, Y. Jiang, and Z. Wang, "Autogan: Neural architecture search for generative adversarial networks," in *ICCV*, 2019.

[96] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation strategies from data," in *CVPR*, 2019.

[97] H. Cai, L. Zhu, and S. Han, "Proxylessnas: Direct neural architecture search on target task and hardware," *ICLR*, 2018.

[98] H. Jin, Q. Song, and X. Hu, "Auto-keras: An efficient neural architecture search system," in *KDD*, 2019.

[99] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," *ICLR*, 2019.

[100] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Smash: one-shot model architecture search through hypernetworks," *ICLR*, 2018.

[101] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive neural architecture search," in *ECCV*, 2018.

[102] M.-h. Oh and G. Iyengar, "Sequential anomaly detection using inverse reinforcement learning," in *KDD*, 2019.

[103] G. Pang, A. v. d. Hengel, C. Shen, and L. Cao, "Deep reinforcement learning for unknown anomaly detection," *arXiv preprint arXiv:2009.06847*, 2020.

[104] K.-H. Lai, D. Zha, Y. Li, and X. Hu, "Dual policy distillation," *arXiv preprint arXiv:2006.04061*, 2020.

[105] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, 1992.

[106] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *ICML*, 2015.

[107] M. Fortunato, C. Blundell, and O. Vinyals, "Bayesian recurrent neural networks," *ICLR*, 2017.

[108] M. Ghavamzadeh, S. Mannor, J. Pineau, A. Tamar, *et al.*, "Bayesian reinforcement learning: A survey," *Foundations and Trends® in Machine Learning*, 2015.

[109] Y. Sun, F. Gomez, and J. Schmidhuber, "Planning to be surprised: Optimal bayesian exploration in dynamic environments," in *AGI*, 2011.

[110] J. Oh and et al, "Self-imitation learning," *ICML*, 2018.

[111] W. Zhang, J. Sun, and X. Tang, "Cat dataset." `https://archive.org/details/CAT_DATASET`, 2009.

[112] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998.

[113] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[114] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," tech. rep., Citeseer, 2009.

[115] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[116] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection," in *CVPR*, 2019.

[117] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *ICLR*, 2017.

[118] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," *ICLR*, 2017.

[119] P. Bergmann, S. Löwe, M. Fauser, D. Sattlegger, and C. Steger, "Improving unsupervised defect segmentation by applying structural similarity to autoencoders," *arXiv preprint arXiv:1807.02011*, 2018.

[120] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *IPMI*, 2017.

[121] P. Napoletano, F. Piccoli, and R. Schettini, "Anomaly detection in nanofibrous materials by cnn-based self-similarity," *Sensors*, 2018.

[122] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *JMLR*, 2012.

[123] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *ICML*, 2006.

[124] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, 2006.

[125] C. D. Manning, C. D. Manning, and H. Schütze, *Foundations of statistical natural language processing*. 1999.

[126] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets," *PloS one*, 2015.

[127] T. DeVries and G. W. Taylor, "Learning confidence for out-of-distribution detection in neural networks," *ICLR*, 2018.

[128] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[129] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," 2012.

[130] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *ICCV*, 2017.

[131] M. Du, S. Pentyala, Y. Li, and X. Hu, "Towards generalizable deepfake detection with locality-aware autoencoder," in *CIKM*, 2020.

[132] Y. Li, Z. Chen, D. Zha, M. Du, D. Zhang, H. Chen, and X. Hu, "Interpretable time-series representation learning with multi-level disentanglement," *arXiv preprint arXiv:2105.08179*, 2021.

[133] F. Yang, Z. Zhang, H. Wang, Y. Li, and X. Hu, "Xdeep: An interpretation tool for deep neural networks," *arXiv preprint arXiv:1911.01005*, 2019.

[134] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *ICLR*, 2015.

[135] Y. Li, X. Huang, J. Li, M. Du, and N. Zou, "Specae: Spectral autoencoder for anomaly detection in attributed networks," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019.

[136] Y. Li, N. Liu, J. Li, M. Du, and X. Hu, "Deep structured cross-modal anomaly detection," in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019.

[137] X. Huang, Q. Song, Y. Li, and X. Hu, "Graph recurrent networks with attributed random walks," in *KDD*, 2019.

[138] Y. Li, D. Zha, N. Zou, and X. Hu, "Pyodds: An end-to-end outlier detection system," *arXiv preprint arXiv:1910.02575*, 2019.

[139] Y. Zhao and et al, "Pyod: A python toolbox for scalable outlier detection," *JMLR*, 2019.

[140] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *NIPS*, 2011.

[141] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation*. Springer Science & Business Media, 2001.

[142] N. Hansen, "The cma evolution strategy: a comparing review," in *Towards a new evolutionary computation*, 2006.

[143] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, 2017.

[144] A. Lavin and S. Ahmad, "Evaluating real-time anomaly detection algorithms–the numenta anomaly benchmark," in *ICML*, 2015.

[145] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recognition Letters*, 2003.

[146] C. C. Aggarwal, "Outlier analysis," in *Data mining*, Springer, 2015.

[147] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *ICDM*, 2008.

[148] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *ACM Sigmod Record*, 2000.

[149] B. Schölkopf and et.al, "Estimating the support of a high-dimensional distribution," *Neural computation*, 2001.

[150] M.-L. Shyu and et al, "A novel anomaly detection scheme based on principal component classifier,"

[151] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "Outlier detection in axis-parallel subspaces of high dimensional data," in *PAKDD*, 2009.

[152] P. Malhotra and et al, "Long short term memory networks for anomaly detection in time series," Presses universitaires de Louvain, 2015.

[153] P. Malhotra and et al, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.

[154] I. Ahmed, T. Galoppo, X. Hu, and Y. Ding, "Graph regularized autoencoder and its application in unsupervised anomaly detection," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 01, pp. 1–1, 2021.

[155] A. Hore and D. Ziou, "Image quality metrics: Psnr vs. ssim," in *2010 20th international conference on pattern recognition*, pp. 2366–2369, IEEE, 2010.