Self-Supervised Anomaly Detection with Neural Transformations

Thesis approved by
the Department of Computer Science
University of Kaiserslautern-Landau
for the award of the Doctoral Degree
Doctor of Natural Sciences (Dr.rer.nat.)

to

Chen Qiu

Date of defense:	April 13th, 2023
Dean:	Prof. Dr. Christoph Garth
Ph.D. committee chair:	Prof. Dr. Karsten Berns
Reviewer:	Prof. Dr. Marius Kloft
Reviewer:	Prof. Dr. Stephan Mandt
Reviewer:	Dr. Maja Rudolph

DE-386

Acknowledgments

It is not an easy task to complete a PhD during the time of COVID-19. Fortunately, I have met great advisors, colleagues, and friends on this journey. I could not have undertaken this thesis without you. First and foremost, I would like to express my sincere gratitude to my exceptional doctoral advisors, Maja Rudolph, Marius Kloft, and Stephan Mandt.

I would like to express my deepest gratitude to Maja, my best friend in research. You initially introduced me to machine learning research and are always a role model to me. I have learned from you not only how to conduct valuable research and also how to support people. Thank you very much for sharing your creative ideas, teaching me scientific writing, and supporting me in both good and bad times. Our extensive and insightful discussions inspired every project of mine in the PhD. I can't imagine without you how tough everything would be.

I am extremely grateful to Marius of TU Kaiserslautern. Thanks for sharing your immense knowledge on anomaly detection in our discussions. With your infectious enthusiasm for both research and life, you have created an open and stimulating environment in our machine learning group, where I met many interesting and smart guys. I am grateful to have the opportunity to be involved in many interesting projects and get advice from Rodrigo Augusto da Silva Alves and Lukas Ruff in writing the thesis. I am also truly glad to have the retreats and hiking with you.

I am also deeply indebted to Stephan of UC Irvine. I am grateful to you for being my external supervisor and in particular for sharing your knowledge and insights from the statistics perspective. I am also thankful to you for initializing many enjoyable and fruitful collaborations and widening my academic social network. I would like to extend my sincere thanks to our fantastic colleagues at UC Irvine, Aodong Li, and Padhraic Smyth. Without your diligent work, some of the publications or submissions would not be possible. It is unbelievable that we have never met each other in person. I am really looking forward to visiting you guys once I am in the U.S.

Also, many thanks also go to all my colleagues at Bosch. Your support made the work and life easier in this hard time. Special thanks go to Manuel Nonnenmacher and Katharina Ott. Thank you for giving valuable suggestions for this thesis. I would be remiss in not mentioning Tim Schneider, whom I was pleased to supervise in his study. Additionally, I would like to convey my heartfelt appreciation to Robert Bosch GmbH, which offered a relaxed research environment with plenty of freedom.

Lastly, I owe my dearest gratitude to my girlfriend Jin and my parents. You are my strongest backing in this tough time. Many thanks to you for supporting me and being by my side with patience all the way. Love you forever.

Abstract

From industrial fault detection to medical image analysis or financial fraud prevention: Anomaly detection—the task of identifying data points that show significant deviations from the majority of data—is critical in industrial and technological applications. For efficient and effective anomaly detection, a rich set of semantic features are required to be automatically extracted from the complex data. For example, many recent advances in image anomaly detection are based on self-supervised learning, which learns rich features from a large amount of unlabeled complex image data by exploiting data augmentations. For image data, predefined transformations such as rotations are used to generate varying views of the data. Unfortunately, for data other than images, such as time series, tabular data, graphs, or text, it is unclear what are suitable transformations. This becomes an obstacle to successful self-supervised anomaly detection on other data types.

This thesis proposes *Neural Transformation Learning*, a self-supervised anomaly detection method that is applicable to general data types. In contrast to previous methods relying on hand-crafted transformations, neural transformation learning learns the transformations from data and uses them for detection. The key ingredient is a novel objective that encourages learning diverse transformations while preserving the relevant semantic content of the data. We prove theoretically and empirically that it is more suited than existing objectives for transformation learning.

We also introduce the extensions of neural transformation learning for anomaly detection within time series and graph-level anomaly detection. The extensions combine transformation learning and other learning paradigms to incorporate vital prior knowledge about time series and graph data. Moreover, we propose a general training strategy for deep anomaly detection with contaminated data. The idea is to infer the unlabeled anomalies and utilize them for updating parameters alternatively. In setups where expert feedback is available, we present a diverse querying strategy based on the seeding algorithm of K-means++ for active anomaly detection.

Our extensive experiments and analysis demonstrate that neural transformation learning achieves remarkable and robust anomaly detection performance on various data types. Finally, we outline specific paths for future research.

Contents

1 Introduction and Overview

Anomaly detection is the task of automatically identifying instances that significantly deviate from the majority of the data. It is critical in a variety of applications from various domains, including intrusion detection in cybersecurity [1, 2], fraud detection in finance [3, 4], industrial fault detection [5, 6], and medical diagnosis [7, 8]. Anomaly detection has been actively studied in various research communities for several decades. The classical anomaly detection methods include robust Principal Component Analysis (PCA) [9, 10], Kernel Density Estimation (KDE) [11, 12], One-Class Support Vector Machine (OCSVM) [13], Local Outlier Factor (LOF) [14], and Isolation Forest (IF) [15, 16]. These shallow methods perform well on low-dimensional data, and their success relies on effective feature engineering.

As data becomes more complex and the size of the dataset goes larger, deep anomaly detection has been rapidly developed in the past years, where the features are learned from data automatically [17]. Related work on deep anomaly detection includes deep autoencoder variants [18, 19, 20], deep one-class classification [21, 22, 23], multi-sphere deep one-class classification [23, 24, 25, 26], deep generative models [27, 28], deep distance-based methods [29, 30], and outlier exposure [31].

Although deep neural networks are universal approximators [32] and have shown impressive performance on various tasks, such as image classification [33], time series analysis [34], and natural language processing [35], it is universally acknowledged that deep learning algorithms are data-hungry [36]. In anomaly detection, the anomalies are often very rare, and the dataset is most of the time unlabeled. The imbalance between normal and abnormal samples and the scarcity of labeled data make the learning of deep anomaly detectors difficult. Many existing deep anomaly detection methods do not learn rich semantic features as supervised learning methods do [37]. Currently, this becomes one of the major bottlenecks inhibiting the improvement of deep anomaly detection methods.

Self-supervised methods are proposed to learn useful features from unlabeled datasets efficiently. The model parameters and features are learned from optimizing the model to solve the designed pretext tasks with data augmentations. The learned features significantly improve the model performance on various downstream tasks.

Recently, there has been a surge of interest in developing self-supervised approaches for anomaly detection. Self-supervised anomaly detection has led to drastic improvements in detection accuracy [37, 38, 39, 40, 41, 42]. Many recent advances in self-supervised anomaly detection rely on the paradigm of data augmentation. Most of the existing self-supervised anomaly detection methods focus on image data since we know what effective data transformations for image data are. However, for data other than images, such as time series, tabular data, graphs, or text, it is much less well-known which transformations are useful, and it is hard to design these transformations manually. It is not clear how to generalize the success of self-supervised anomaly detection on images to other data types.

Motivated by this, this thesis proposes a self-supervised anomaly detection method with learnable data transformations that is applicable to general data types and applications. The approach, which we call *Neural Transformation Learning*, learns a variety of diverse and semantically meaningful transformations from data and exploits the learned transformations for anomaly detection.

1.1 The Thesis

This thesis studies how to apply self-supervised anomaly detection to general data types and applications. We summarize the main contributions, the list of publications, and the organization of the thesis in the following.

1.1.1 Contributions

The main contributions and findings of this thesis are the following:

- We introduce Neural Transformation Learning (NTL), a self-supervised anomaly detection method with learnable transformations. The key idea is to learn a variety of diverse and semantically meaningful transformations. This unleashes the power of self-supervised anomaly detection to general data types. To do so, we propose a novel contrastive loss that encourages the transformed samples to share semantic information with their original form while different views are easily distinguishable. We theoretically demonstrate that the proposed loss is better suited than existing self-supervised losses for transformation learning and anomaly detection. Our extensive empirical study finds that NTL achieves impressive anomaly detection results on various data types.

- We introduce an extension of NTL for detecting anomalies within time series, called Local Neural Transformations (LNT), which learns local transformations at each time step from data and produces an anomaly score for each time step.

The key ingredient is a novel training objective combining time series representation learning and transformation learning. We demonstrate theoretically and empirically that both learning paradigms complement each other, yielding a strong time series anomaly detection model.

- We introduce two extensions of NTL for graph-level anomaly detection, which refers to detecting entire abnormal graphs in a set of graphs. One-Class Graph Transformation Learning (OCGTL) combines the best of deep one-class classification and transformation learning through a joint loss with two loss terms, respectively. Multi-view One-Class Classification (MOCC) generalizes deep one-class classification by involving multiple learnable transformations. Both extensions raise the graph-level anomaly detection accuracy significantly.
- We introduce Latent Outlier Exposure (LOE), a general strategy for training deep anomaly detection models (including NTL) on unlabeled contaminated data. The idea is to jointly infer binary labels to each datum (normal vs. anomalous) while updating the model parameters. We use a combination of two losses that share parameters: one for the normal and one for the anomalous data. We then proceed with block coordinate updates on the parameters and the most likely (latent) labels. Our experiments reveal improved performance over established baselines on image, tabular, and video data.
- We introduce Active Latent Outlier Exposure (ALOE): an active learning approach for deep anomaly detection models, including NTL. ALOE relies on a diversified querying strategy based on the seeding algorithm of K-means++ and a combination of two losses for queried and unqueried samples. Based on a simple heuristic for weighting the losses relative to each other and by estimating the unknown contamination rate from queried samples, we were able to make our approach free of its most important hyperparameters. We showed on a variety of datasets from different domains that the approach results in a new state-of-the-art in active anomaly detection.

Overall, the contributions and findings above demonstrate that NTL is a powerful and robust self-supervised anomaly detection method that is applicable to various data types and applications.

1.1.2 List of Publications

The primary contributions and findings of this thesis are based on the following peer-reviewed publications:

Chen Qiu, Timo Pfrommer, Marius Kloft, Stephan Mandt, and Maja Rudolph. Neural Transformation Learning for Anomaly Detection beyond Images. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of Proceedings of Machine Learning Research, pages 8703–8714. PMLR, 18–24 Jul 2021.

Tim Schneider, **Chen Qiu**, and Maja Rudolph. Anomalous Region Detection in Time Series with Local Neural Transformations. In *ICML 2021 Workshop: Self-Supervised Learning for Reasoning and Perception*, 2021.

Chen Qiu, Marius Kloft, Stephan Mandt, and Maja Rudolph. Raising the Bar in Graph-level Anomaly Detection. In *In Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, IJCAI-22, pages 2196–2203. International Joint Conferences on Artificial Intelligence Organization, 7 2022. Main Track.

Chen Qiu*, Aodong Li*, Marius Kloft, Maja Rudolph, and Stephan Mandt. Latent Outlier Exposure for Anomaly Detection with Contaminated Data. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of Proceedings of Machine Learning Research, pages 18153–18167. PMLR, 17–23 Jul 2022.

Aodong Li*, **Chen Qiu***, Padhraic Smyth, Marius Kloft, Stephan Mandt, and Maja Rudolph. Diverse Querying Improves Deep Active Anomaly Detection. *Preprint (under review)*, 2022.

Chen Qiu, Marius Kloft, Stephan Mandt, and Maja Rudolph. Self-Supervised Anomaly Detection with Neural Transformations. *Preprint (under review)*, 2022.

We note that all co-authors of these works have agreed to borrow ideas, figures, and results from the works above for this thesis.

1.1.3 Organization of the Thesis

This thesis comprises three main chapters:

Chapter 2 (Neural Transformation Learning) introduces NTL, an end-to-end self-supervised anomaly detection approach for general data types. We first present the algorithm for learning diverse and semantically meaningful transformations from data. We then provide theoretical results proving that NTL is better suited for transformation learning than existing work. In the experimental evaluation, we find that NTL achieves superior detection accuracy on various data types, including time series, images, tabular data, and text.

*Equal contribution

Chapter 3 (Extensions of Neural Transformation Learning) introduces extensions that improve NTL for specific applications, including anomaly detection within time series and graph-level anomaly detection. We first introduce LNT for detecting anomalies within time series, which learns local transformations at each time step by combining transformation learning and representation learning. We show that LNT outperforms many strong baselines on challenging time series anomaly detection tasks. We then introduce OCGTL and MOCC for graph-level anomaly detection, which refers to detecting entire abnormal graphs in a set of graphs. Both OCGTL and MOCC combine NTL and deep one-class classification. In the experimental evaluation, we show that OCGTL and MOCC raise the bar in graph-level anomaly detection.

Chapter 4 (Neural Transformation Learning with Contaminated Data) introduces two learning paradigms for deep anomaly detection that improve the robustness of NTL to contaminated training data. We first present LOE for training deep anomaly detectors on unlabeled contaminated data. LOE jointly infers the anomaly labels and learns the model parameters during training. The experimental evaluation demonstrates its benefits in learning deep anomaly detectors with unlabeled contaminated data. We then introduce ALOE for setups where expert feedback is available in training. ALOE queries samples that are diverse in the embedding space and uses a combination of two losses for queried and unqueried samples. We empirically show that ALOE achieves a new state-of-the-art in active anomaly detection.

We conclude the thesis and provide detailed paths for future research in Chapter 5. Before we turn to the main chapters of the thesis, we complete this introduction and overview with a survey of the existing self-supervised anomaly detection methods.

1.2 Background and Challenges

This section provides an overview of self-supervised anomaly detection. In Section 1.2.1, we briefly review the development of self-supervised learning. In Section 1.2.2, we discuss the existing self-supervised anomaly detection methods and categorize them into end-to-end methods and two-stage methods. We finally highlight the notable challenges in self-supervised anomaly detection.

1.2.1 Self-Supervised Learning in a Nutshell

Self-supervised learning learns semantic features from unlabeled data without time-consuming and expensive data annotations [43]. Self-supervised learning typically designs pretext tasks with automatically generated pseudo labels as the learning problem for the model. The model is optimized to solve the pretext task. Effective pretext tasks ensure that meaningful semantic features are learned during the training. For example, in the work of learning representations by predicting image rotations [44], images are rotated with four different angles to generate data augmentations. The rotation angles serve as pseudo labels. In training, the neural networks are optimized to predict the rotation angle of the input image. The training forces the networks to learn semantic features useful for solving the task. The learned representations are then used in downstream tasks that require semantic features, such as image classification, object detection, and semantic segmentation.

Following a similar pipeline, a variety of self-supervised learning methods have been developed for various data types. For images, the networks are trained on pretext tasks, such as patch prediction [45], image colorization [46], image inpainting [47], solving jigsaw puzzles [48, 49], cross-channel prediction [50], or rotation prediction [44]), to learn semantic visual features. Temporal structure-based pretext tasks are shown to be effective for learning video representations [51, 52, 53, 54]. For learning rich language representations, BERT [55] and its variants [56, 57] train transformers [58] to predict the randomly masked words. Xie et al. [59] provide a unified review of self-supervised learning on graphs, including generation-based, property prediction-based, and contrastive-based methods.

Contrastive-based self-supervised methods relying on the InfoMax principle [60, 61] received a lot of attention recently. These methods are trained on the task to maximize the mutual information between the data and their context [62, 63, 64] or between different "views" of the data [65]. Computing the mutual information in these settings is often intractable, and various approximation schemes and bounds have been introduced [66]. By using noise contrastive estimation [67, 68] to bound mutual information, Oord et al. [62] bridge the gap between contrastive losses for mutual information-based representation learning and the use of contrastive losses in discriminative methods for representation learning [65, 69, 70, 71, 72, 73]. Moreover, Grill et al. [74], Chen and He [75] show that just aligning the different views of the same sample with Siamese network can learn meaningful representations even without using negative samples.

Self-supervised learning learns rich semantic features useful for various downstream tasks, including image classification [72], object detection [76], machine translation [77], and also anomaly detection. Next, we review the existing self-supervised anomaly detection methods.

1.2.2 Self-Supervised Anomaly Detection

Self-supervised methods have also been successfully applied to anomaly detection. The essence of an anomaly detection algorithm is to produce a score function

$s(\boldsymbol{x}) : \mathcal{X} \to \mathbb{R}$, which measures the degree of abnormality of each sample in the data space $\mathcal{X}$. The score function is used at test time to detect anomalies by evaluating whether the score is above a threshold,

$$\tilde{y} = \begin{cases} 1, & \text{if } s(\boldsymbol{x}) \geq \text{threshold} \qquad \text{meaning } \boldsymbol{x} \text{ abnormal} \\ 0, & \text{otherwise} \qquad \text{meaning } \boldsymbol{x} \text{ normal.} \end{cases} \tag{1.1}$$

The threshold is calibrated for the specific application (e.g., using a labeled validation set). Since the anomalies are often very rare and annotations are not available in general, a data-driven anomaly detection algorithm relies on the unlabeled training dataset $\mathcal{D} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\} \subset \mathcal{X}$ to learn a score function. Here we consider methods that learn the score function or/and its associated model with self-supervised learning. We categorize the self-supervised anomaly detection methods into two types: end-to-end methods and two-stage methods.

1.2.2.1 End-to-End Self-Supervised Anomaly Detection

Self-supervised learning was pioneered for end-to-end anomaly detection by Golan and El-Yaniv [38] and put forward in Hendrycks et al. [39], Wang et al. [40], Tack et al. [42]. They proposed various pretext tasks with *hand-crafted* data transformations for learning model parameters and semantic features. The anomaly score function can be constructed based on the model performance on the pretext task, such as the loss function [31, 38, 40]. The anomalies can also be detected by shallow anomaly detectors (e.g., OCSVM and KDE) built upon the learned representations [42].

In Golan and El-Yaniv [38], Wang et al. [40], a set of hand-crafted image augmentations such as rotations, shifting, and flipping are applied to each input image. A classifier is learned with a multi-class classification loss to predict which transformation has been applied to the input image. At test time, all generated views of the test image are fed into the trained classifier. The prediction accuracy is shown to be an effective anomaly score. Wang et al. [40] studied why the model performance on the pretext task can reflect the normality of samples and proposed the principle of inlier priority. Since most training samples are inliers, the training will promote the model to generalize better to held-out normal samples than anomalies. Therefore, the anomalies have higher loss/score values than the normal samples. Besides that, they empirically show that the transformation prediction-based anomaly detection methods are robust to a contaminated training set.

Hendrycks et al. [39] improve the transformation prediction-based detection by learning a multi-head classifier for transformation prediction. The test images are scored by the model performance on multiple classification tasks. Specifically, the model learns to solve three different tasks: one for predicting rotation, one for predicting vertical shift, and one for predicting horizontal shift. The model has three softmax heads, each for a classification task. By involving three tasks in the learning problem and anomaly detection, the model is more sensitive to abnormal images and achieves better detection accuracy.

Tack et al. [42] first applied contrastive learning [73] to image anomaly detection. Contrastive learning learns representations by aligning different views of the same data and contrasting them to other samples. The anomalies are scored by the cosine similarity of their representations to the nearest training sample's representation and the norm of their representations. They found that the rotated images can serve as synthetic anomalies in training. By contrasting to rotated images, they learn a better anomaly detector than other self-supervised anomaly detection methods. Similarly, Chen et al. [78] utilize negative data augmentations, including rotation, clipping, and color jitter, to simulate novelty-like samples and learn an encoder-decoder framework via contrastive learning for novelty detection. Song et al. [79], Zavrtanik et al. [80] learn anomaly segmentation models following the idea of contrasting to views generated by various hand-crafted negative transformations. Apart from designing negative samples, Goyal et al. [81] learn to generate negative samples for a robust deep one-class classification. The generated negative samples are outside a small hypersphere centered around a normal sample in the embedding space.

The above-mentioned self-supervised anomaly detection methods focus on image data and rely on well-designed image transformations. There are also pretext tasks and data transformations designed specifically for anomaly detection on data types beyond images. For detecting anomalies within time series, Deldari et al. [82] adopt contrastive predictive coding, which learns representations maximizing the mutual information between consecutive time windows [62], while Carmona et al. [83] contrast the representations of two overlapped time series windows and design various types of synthetic time series anomalies. Bergman and Hoshen [41] generalize the transformation prediction-based detection method to tabular data by using random-affine transformations, while Zhang et al. [84] further improve the idea by modeling the feature distribution of normal samples with neural autoregressive flows and enabling an active learning scheme for querying informative samples in training. However, the random-affine transformations are apparently noisy and not optimal for tabular data.

1.2.2.2 Two-Stage Self-Supervised Anomaly Detection

The learned representations in self-supervised learning are useful in various downstream tasks, including anomaly detection [37, 85, 86, 87, 88]. Typically, two-stage self-supervised anomaly detection methods learn a feature extractor with self-supervised learning in the first stage and build an anomaly detector upon the representations in the second stage. For example, in Sehwag et al. [85], the feature extractor is learned by minimizing a contrastive loss [73]. For detection, they first partition the learned representations of training data into clusters. The anomalies are then scored by the Mahalanobis distance to the nearest cluster center in the feature space.

Sohn et al. [37] learn representations that are more suitable for anomaly detection by treating the rotated images as negative samples in contrastive learning. Given the learned representations, the detection can be easily done with shallow anomaly detectors such as OCSVM [13] and KDE [11]. Furthermore, Reiss and Hoshen [87] proposed a mean-shifted contrastive loss for learning representations for the downstream anomaly detection task. The mean-shifted contrastive loss combines the contrastive loss [73] with the center loss [89] to pull the representation of normal samples to a center. The anomalies are scored by the cosine distance of their features to the k-nearest training samples' features. Cho et al. [86] proposed masked contrastive learning for anomaly detection and showed that the learned representations could be even better when additional class information is available.

Similar two-stage frameworks are also proposed to localize the anomalous regions of the image. For example, Li et al. [88] learn an encoder to extract representations for each image patch. The key novelty is the proposed CutPaste augmentations for creating irregular local patterns. The CutPaste augmentation cuts a small rectangular area from a normal sample and then pastes the rotated patch back to an image at a random location. The encoder is trained to predict if the image patch is irregular due to the CutPaste augmentation and therefore learns representations capturing local patterns. In the second stage, a Gaussian density estimator is built upon the patch representations to compute the anomaly score of each patch.

1.2.3 Challenges

We conclude this introduction by highlighting the notable challenges in self-supervised anomaly detection, which also motivate this thesis. Since the transformed views share semantic information (a human still recognizes a rotated cat as a cat), the model starts memorizing salient features of normal samples in learning to solve the pretext tasks and can later be used for anomaly detection. This idea has led to strong anomaly detectors based on either transformation prediction [38, 39, 40] or using representations learned using these views [73] for downstream anomaly detection tasks [37, 85, 87]. We can see that the advances in self-supervised anomaly detection rely on the paradigm of data augmentation.

For applying self-supervised learning to each data type, the domain-specific data transformations are carefully designed by experts [90]. For image data, predefined transformations such as rotations, reflections, and cropping are used to generate various views of the data. Unfortunately, for data other than images, such as time series, tabular data, text, and graphs, it is much less well-known which transformations are useful. Also, the pretext tasks are often specified for one application and are not effective for other tasks. For example, contrasting between sequence windows yields a good model for detecting anomalies within time series [82, 83], but is not appropriate for image anomaly detection. Therefore, one major obstacle to generalizing the principles of self-supervised anomaly detection to other data types and applications is designing proper transformations for various data types.

This motivates one of the main contributions of this thesis, NTL, which follows the principles of self-supervised anomaly detection and learns the transformations from data. In Chapter 2, we will dive into the algorithm, theory, and applications of NTL in self-supervised anomaly detection. NTL is the first method to learn transformations for self-supervised anomaly detection. Thanks to the flexibility of learnable transformations, NTL is the first self-supervised anomaly detection method applicable to various data types, including images, time series, tabular data, and text. In Chapter 3, we will introduce the extensions of NTL. By combining with time series representation learning, NTL detects anomalies within time series effectively. By combining with deep one-class classification, NTL achieves state-of-the-art performance in graph-level anomaly detection.

Besides the generalization of self-supervised anomaly detection methods, one other challenge is the robustness of the methods to data contamination. A common assumption in many (self-supervised) anomaly detection methods is that a “clean” training set (free of anomalies) is available to teach the model the features of normal samples. However, the training set could contain unnoticed anomalies already. For example, a dataset of medical images may already contain cancer images, or datasets of financial transactions could already contain unnoticed fraudulent activities. Naively training an anomaly detector on such data suffers from degraded performance [40, 91]. We address the robustness challenge in Chapter 4. We will introduce a novel training strategy, LOE, that infers unlabeled anomalies while updating the model parameters. Furthermore, we will study the training and query strategies in active anomaly detection and introduce ALOE, which queries diverse and informative samples and optimizes the model on both labeled and unlabeled samples.

2 Neural Transformation Learning

In this chapter, we introduce Neural Transformation Learning (NTL): an end-to-end self-supervised method for anomaly detection with *learnable* transformations. Instead of manually designing data transformations to construct auxiliary prediction tasks that can be used for anomaly detection, we derive a single objective function for both learning useful data transformations and anomaly scoring. The idea is to learn a variety of transformations such that the transformed samples share semantic information with their original form while different views are easily distinguishable.

We first introduce the algorithm and methodology of NTL in Section 2.1. NTL has a set of learnable transformations which are jointly trained on a Deterministic Contrastive Loss (DCL) designed to learn faithful transformations. In Section 2.2, we then provide several theoretical arguments for why DCL is better suited for transformation learning than alternative loss functions. We prove that optimizing existing losses for self-supervised learning w.r.t. the transformations leads to trivial edge-cases, while these edge-cases are not optimal solutions under our objective DCL. Therefore, we can use DCL for transformation learning and anomaly detection. In Section 2.3, we find that our approach achieves impressive results in anomaly detection on various data types. For time series, tabular data, and text, NTL significantly improves the accuracy of anomaly detection. In terms of images, NTL generalizes self-supervised anomaly detection/segmentation to image features and achieves comparable results to state-of-the-art methods.

2.1 Algorithm and Methodology of NTL

NTL learns an anomaly detector which contains $K+1$ *learnable* transformations. Each transformation is a parameterized function (e.g., a neural network) $\mathcal{T}_k^\theta(\cdot) : \mathcal{X} \to \mathcal{Z}$ that maps samples into an embedding space $\mathcal{Z}$. These embeddings include one reference embedding $\mathcal{T}_0^\theta(\boldsymbol{x})$ and K transformed embeddings $\mathcal{T}_k^\theta(\boldsymbol{x})$ for $k = 1, \ldots, K$. We assume that all transformations are learnable, i.e., they can be modeled by any parameterized function whose parameters θ are accessible to gradient-based optimization. A schematic of NTL is in Figure 2.1. Each sample is transformed by a

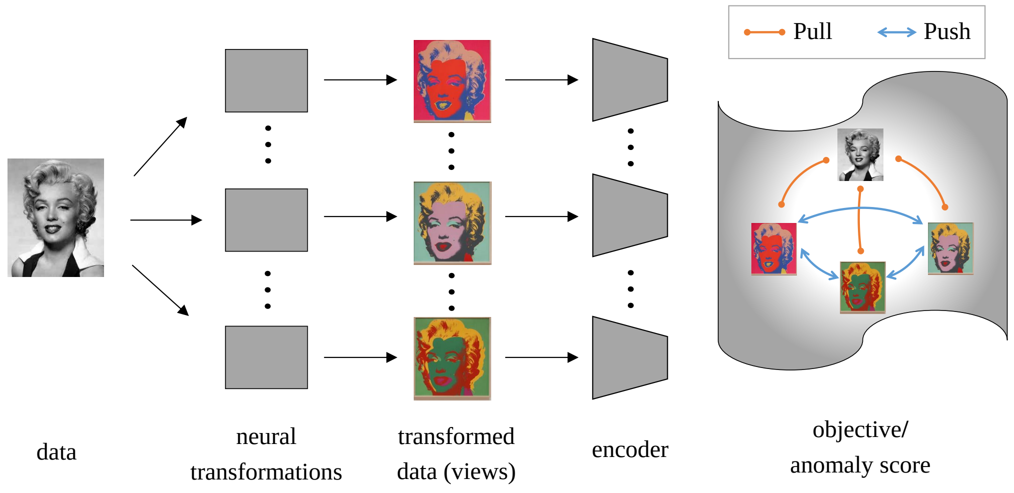


Figure 2.1: NTL is an end-to-end procedure for self-supervised anomaly detection with *learnable* neural transformations. Each sample is transformed by a set of neural transformations and then embedded into a semantic space. The transformations and the encoder are trained jointly with a contrastive objective (Equation (2.2)), which is also used to score anomalies. Note that the figures of Marilyn Monroe [92] are only used for illustration purposes.

set of learnable transformations and then embedded into a semantic space such that (1) all transformed embeddings preserve some semantic information about the input sample, while (2) all transformations remain distinct from another. As follows, we motivate NTL's loss function and its usage in anomaly scoring. For convenience, we list the basic notations in Table 2.1.

2.1.1 Loss Function and Optimization

A key ingredient of NTL is a new loss function that we call Deterministic Contrastive Loss (DCL), explained below. This loss is constructed to encourage the transformations to learn salient features of the training data (which are assumed to belong to the normal class). The DCL encourages each transformed embedding to be similar to the reference embedding while encouraging it to be dissimilar from other transformed embeddings of the same sample. We define a kernel function [93] for measuring the similarity of two embeddings $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{Z}$ as

$$h(\boldsymbol{a}, \boldsymbol{b}) := \exp\left(\frac{\boldsymbol{a}^\top \boldsymbol{b}/\|\boldsymbol{a}\|\|\boldsymbol{b}\|}{\tau}\right), \tag{2.1}$$

Table 2.1: Basic notations

Indices:	
K	number of transformations ($k \in \{1, ..., K\}$)
N	number of data samples ($i \in \{1, ..., N\}$)
$\mathcal{N}$	number of data classes
Variables:	
$\boldsymbol{x}$	data sample
y	ground-truth label of being an anomaly
$\tilde{y}$	predicted label of being an anomaly
$\mathcal{X}$	data space
$\mathcal{Z}$	embedding space
Parameters:	
θ	learnable parameters of the model
τ	temperature controlling the sensitivity of the similarity measure
Functions:	
$\phi_k(\cdot)$	transformation function $\mathcal{X} \to \mathcal{X}$
$\phi_k^\theta(\cdot)$	learnable neural transformation function $\mathcal{X} \to \mathcal{X}$
$\mathcal{T}_k^\theta(\cdot)$	learnable neural transformation function $\mathcal{X} \to \mathcal{Z}$
$f^\theta(\cdot)$	trainable encoder network $\mathcal{X} \to \mathcal{Z}$
$h(\cdot,\cdot)$	similarity function between two variables $\mathcal{Z} \to \mathbb{R}$
$\ell(\cdot)$	loss function $\mathcal{X} \to \mathbb{R}$
$s(\cdot)$	anomaly score function $\mathcal{X} \to \mathbb{R}$

where τ denotes a temperature parameter. By plugging all transformations and averaging over all data samples, DCL is defined using the function $h(\boldsymbol{a}, \boldsymbol{b})$ as

$$\mathcal{L}_{\mathrm{NTL}}^\theta = \frac{1}{N}\sum_{i=1}^{N} \ell(\boldsymbol{x}_i; \theta) \quad \text{with } \ell(\boldsymbol{x}; \theta) := -\sum_{k=1}^{K} \log \frac{h(\mathcal{T}_k^\theta(\boldsymbol{x}), \mathcal{T}_0^\theta(\boldsymbol{x}))}{\sum_{l \in \{0,\dots,K\}/\{k\}} h(\mathcal{T}_k^\theta(\boldsymbol{x}), \mathcal{T}_l^\theta(\boldsymbol{x}))}. \tag{2.2}$$

Each summand selects one transformed embedding as the anchor. The reference embedding is the positive sample, and the other transformed embeddings are the negative samples. Intuitively, the term in the nominator pulls the transformed embeddings close to the reference embedding, encouraging the transformed embeddings to preserve relevant semantic information of the reference embedding. The denominator pushes all transformed embeddings away from each other, thereby encouraging diverse transformations.

The parameters of all transformations θ are optimized jointly with stochastic gradient descent or its variants (e.g., Adam [94]). The optimal parameters θ^* are computed by minimizing Equation (2.2) on the training samples as

$$\theta^* = \arg\min_\theta -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\log\frac{h(\mathcal{T}_k^\theta(\boldsymbol{x}_i), \mathcal{T}_0^\theta(\boldsymbol{x}_i))}{\sum_{l\in\{0,\ldots,K\}/\{k\}} h(\mathcal{T}_k^\theta(\boldsymbol{x}_i), \mathcal{T}_l^\theta(\boldsymbol{x}_i))}. \tag{2.3}$$

This training objective leads to neural transformations that produce diverse views of each sample. Next, we describe how NTL is used to detect anomalies.

2.1.2 Score Function

One advantage of our approach over other methods is that our loss function is also our anomaly score. We define an anomaly score $s(\boldsymbol{x})$ as

$$s(\boldsymbol{x}) := \ell(\boldsymbol{x}; \theta). \tag{2.4}$$

By minimizing the DCL (Equation (2.2)), we minimize the anomaly score for training samples (inliers). The transformations learn to highlight salient features of the data such that a low loss value can be achieved. After training, samples from the normal class have low anomaly scores, while anomalies are handled less well by the model and thus have high scores. The higher the anomaly score, the more likely that a sample is an anomaly. Unlike most other contrastive losses in representation learning [68, 73] and image anomaly detection [37, 42], the negative samples are not drawn from a noise distribution (e.g., other samples in the minibatch) but constructed deterministically from $\boldsymbol{x}$ in our loss and score function. Dependence on the minibatch for negative samples would need to be accounted for at test time. Drawing negative samples from the test data is biased, and using the training data is infeasible in practice. In contrast, the deterministic nature of our score function makes it a simple choice for anomaly detection.

This concludes the proposed method NTL, an end-to-end procedure for transformation learning and anomaly detection. The detailed training process and test process of NTL are in Algorithm 1. We stress that it is simple and effective without the need of any additional regularization. However, it is not trivial to see why NTL is suited for transformation learning and anomaly detection. To this end, we analyze the theoretical properties of NTL.

2.2 Theoretical Properties of NTL

We provide theoretical considerations on what makes the DCL suitable for transformation learning, especially in comparison to other popular self-supervised losses. We start by introducing two requirements that allow machine learning algorithms to learn useful transformations. We then show that the requirements are not met for

Algorithm 1: Algorithm of NTL

```
Data: training dataset $\mathcal{D}$, test data $\boldsymbol{x}_{\text{test}}$
Model: neural transformations $\mathcal{T}^{\theta}_{0,\ldots,K}$
// Training process
foreach epoch do
    foreach Mini-batch $\mathcal{M} \subset \mathcal{D}$ do
        Transform each training sample $\mathcal{T}^{\theta}_{0}(\boldsymbol{x}), \cdots, \mathcal{T}^{\theta}_{K}(\boldsymbol{x}) \leftarrow \boldsymbol{x} \in \mathcal{M}$
        Minimize the loss $\mathcal{L}^{\theta}_{\text{NTL}}$ in Equation (2.2) to update $\theta$
    end
end
// Test process
Transform the test data $\mathcal{T}^{\theta}_{0}(\boldsymbol{x}_{\text{test}}), \cdots, \mathcal{T}^{\theta}_{K}(\boldsymbol{x}_{\text{test}}) \leftarrow \boldsymbol{x}_{\text{test}}$
Calculate the anomaly score according to $s(\boldsymbol{x}_{\text{test}})$ in Equation (2.4)
```

existing related self-supervised methods, preventing these loss functions from being useful for transformation learning. In contrast, we demonstrate that NTL can learn useful transformations that satisfy both requirements and lead to a powerful and flexible self-supervised anomaly detection method.

2.2.1 When Can Transformations be Learned from Data?

We discuss the fundamentals for transformation learning in the following. Data transformations play an important role in auxiliary tasks for self-supervised learning. Here we provide considerations for what properties make the transformations useful. These considerations guide our analysis of various self-supervised losses to decide if they are suitable for learning transformations from data.

Current transformation prediction methods are based on the following human intuition: we can recognize and thereby describe transformations applied to data we know. For example, we can recognize a rotated car or tree. However, assume an image of a strange shape that we have never encountered before: since we would not recognize it, we could also not tell if a transformation (e.g., rotation) has been applied to it. This explains why transformation prediction on normal data is easier than predicting which transformation has been applied to an anomaly.

It is natural to ask how transformations are typically chosen. On the one hand, the transformations typically do not render the input unrecognizable. A rotated/blurred/cropped cat is still recognizable as a cat. Training a transformation predictor with such transformations gives us an effective feature extractor. On the other hand, when applying more transformations, the transformation prediction task becomes harder, and the model needs to become more sensitive to the characteristic features of the data to solve the learning task well. To make this task meaningful, one typically chooses a diverse set of transformations.

Since the machine learning framework of transformation prediction draws on the same principles, one may wonder which properties one can define as practical necessities for *learnable* transformations. We argue that loss functions for learning useful transformations for downstream tasks need to meet two fundamental goals:

Requirement 1 (Semantics)**.** Learned transformations should produce views that share relevant semantic information with the original data.

Requirement 2 (Diversity)**.** Learned transformations should produce diverse views of each sample.

A valid loss function for transformation learning should avoid solutions that violate either of these requirements. There are plenty of transformations that would violate Requirement 1 or Requirement 2. For example, a constant transformation that does not depend on $\boldsymbol{x}$ would violate the semantics requirement, whereas identical transformations that produce the same views violate the diversity requirement.

We argue thus that for self-supervised anomaly detection, the learned transformations need to negotiate the trade-off between semantics and diversity with the two examples as edge cases on a spectrum of possibilities. Without semantics, i.e., without dependence on the data, an anomaly detection method cannot decide whether a new sample is normal or anomalous. And without variability in transformations, the benefit from self-supervised learning is gone in anomaly detection.

2.2.1.1 Are Existing Losses Suited for Transformation Learning?

Before discussing NTL in detail, we compare the approach with two other self-supervised approaches using data transformations. We will prove that these popular frameworks do not meet one of the two goals discussed above and therefore do not allow the data transformations to be *learned*. These frameworks are discussed next.

1. The first approach is the transformation prediction approach by Golan and El-Yaniv [38]. It requires a set of *hand-crafted* transformations $\{\phi_1, ..., \phi_K \mid \phi_k(\cdot) : \mathcal{X} \to \mathcal{X}\}$. Given a transformed view $\phi_k(\boldsymbol{x}) \in \mathcal{X}$, a classifier $f^\theta(\cdot) : \mathcal{X} \to \mathbb{R}^K$ that outputs K values $f^\theta(\phi_k(\boldsymbol{x}))_1 \dots f^\theta(\phi_k(\boldsymbol{x}))_K$ proportional to the log-probabilities of which transformation was used to produce the view. The transformation prediction loss is a softmax classification loss formulated with $\mathcal{T}_k^\theta = f^\theta \circ \phi_k$, where $\circ$ denotes the composition of functions.

$$\mathcal{L}_P^\theta := -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} \log \frac{\exp \mathcal{T}_k^\theta(\boldsymbol{x}_i)_k}{\sum_{l=1:K} \exp \mathcal{T}_k^\theta(\boldsymbol{x}_i)_l}. \tag{2.5}$$

2. We also consider Chen et al. [73], who define a contrastive loss on each minibatch of data $\mathcal{M} \subset \mathcal{D}$. For each gradient step, they sample a minibatch and two *hand-crafted* transformations ϕ_0, ϕ_1 from the family of transformations, which are applied to all the samples. Here $f^\theta(\cdot) : \mathcal{X} \to \mathcal{Z}$ is an encoder that maps data to the embedding space. Using $\mathcal{T}_k^\theta = f^\theta \circ \phi_k$, the loss function is given by

$$\mathcal{L}_C^\theta := \tag{2.6}$$
$$-\frac{1}{|\mathcal{M}|}\sum_{i=1}^{|\mathcal{M}|}\sum_{k=0}^{1}\log\frac{h(\mathcal{T}_k^\theta(\boldsymbol{x}_i), \mathcal{T}_{1-k}^\theta(\boldsymbol{x}_i))}{\sum_{\boldsymbol{x}_j \in \mathcal{M}/\{\boldsymbol{x}_i\}} h(\mathcal{T}_k^\theta(\boldsymbol{x}_i), \mathcal{T}_k^\theta(\boldsymbol{x}_j)) + \sum_{\boldsymbol{x}_j \in \mathcal{M}} h(\mathcal{T}_k^\theta(\boldsymbol{x}_i), \mathcal{T}_{1-k}^\theta(\boldsymbol{x}_j))}$$

With hand-crafted, fixed transformations, these losses produce excellent anomaly detectors for images. Specifically, Golan and El-Yaniv [38], Wang et al. [40], Tack et al. [42] use the self-supervised loss function also as the score function to detect anomalies in an end-to-end manner. Sohn et al. [37], Sehwag et al. [85] present a two-stage framework for anomaly detection. In the first stage, they learn high-level data representations by minimizing the contrastive loss. In the second stage, a shallow anomaly detection method, such as an OCSVM [13], or a distance metric, such as Mahalanobis distance, is applied to the representations from the first stage. Since it is not always straightforward to design transformations for new application domains, we study their suitability for *learning* data transformations.

We prove that $\mathcal{L}_P^\theta$ and $\mathcal{L}_C^\theta$ are less well-suited for transformation learning than $\mathcal{L}_{\text{NTL}}^\theta$ (Equation (2.2)). We start by arguing that $\mathcal{L}_P^\theta$ has a trivial solution when trying to optimize the transformations. For this reason, the transformations have to be hand-crafted, and the transformation prediction approach can only be used for data where enough intuition exists to do so.

Proposition 1. The 'constant' edge-case $\mathcal{T}_k^\theta(\boldsymbol{x}) = \sigma \boldsymbol{c}_k$ for $k = 1, \ldots, K$, where $\boldsymbol{c}_k$ is a one-hot vector encoding the k^{th} position (i.e., $\boldsymbol{c}_{k,k} = 1$), tends towards the minimum of $\mathcal{L}_P^\theta$ (Equation (2.5)) as the constant σ goes to infinity.

Proof. As a negative log probability, $\mathcal{L}_P^\theta \geq 0$ is lower bounded by 0. We want to show that with $\mathcal{T}_k^\theta(\boldsymbol{x}) = \sigma \boldsymbol{c}_k$, where $\boldsymbol{c}_k$ is a one-hot vector and σ is a constant, $\mathcal{L}_{\text{NTL}}^\theta$ goes to 0 as σ goes to infinity. Plugging $\mathcal{T}_k^\theta(\boldsymbol{x}) = \sigma \boldsymbol{c}_k$ into $\mathcal{L}_P^\theta$ and taking the limit yields

$$\lim_{\sigma\to\infty}\mathcal{L}_P^\theta = \lim_{\sigma\to\infty} -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\log\frac{\exp\sigma}{\exp\sigma + K - 1} = \lim_{\sigma\to\infty} -K\log\frac{\exp\sigma}{\exp\sigma + K - 1}$$
$$= \lim_{\sigma\to\infty} -K\sigma + K\log(\exp\sigma + K - 1) = 0.$$

Therefore, Proposition 1 is proven. □

The constant transformations violate the semantics requirement and therefore cannot be used for anomaly detection. Proposition 1 implies that $\mathcal{L}_P^\theta$ cannot be used to learn transformations. An alternative popular self-supervised loss $\mathcal{L}_C^\theta$ is not suitable for transformation learning either. We next show that $\mathcal{L}_C^\theta$ is trivially optimized by identity transformations.

Proposition 2. The 'identity' edge-case $\phi_0(\boldsymbol{x}) = \phi_1(\boldsymbol{x}) = x$ with an adequate encoder is a minimizer of $\mathcal{L}_C^\theta$ (Equation (2.6)).

Proof. $\mathcal{L}_C^\theta$ can be written as $\mathcal{L}_C^\theta = \mathcal{L}_{\text{alignment}}^\theta + \mathcal{L}_{\text{uniformity}}^\theta$, where

$$\mathcal{L}_{\text{alignment}}^\theta = -\frac{1}{|\mathcal{M}|}\sum_{i=1}^{|\mathcal{M}|}\sum_{k=0}^{1}\log h(\mathcal{T}_k^\theta(\boldsymbol{x}_i), \mathcal{T}_{1-k}^\theta(\boldsymbol{x}_i)),$$

$$\mathcal{L}_{\text{uniformity}}^\theta = \frac{1}{|\mathcal{M}|}\sum_{i=1}^{|\mathcal{M}|}\sum_{k=0}^{1}\log \sum_{\boldsymbol{x}_j\in\mathcal{M}/\{\boldsymbol{x}_i\}} h(\mathcal{T}_k^\theta(\boldsymbol{x}_i), \mathcal{T}_k^\theta(\boldsymbol{x}_j)) + \sum_{\boldsymbol{x}_j\in\mathcal{M}} h(\mathcal{T}_k^\theta(\boldsymbol{x}_i), \mathcal{T}_{1-k}^\theta(\boldsymbol{x}_j)).$$

A sufficient condition of $\min(\mathcal{L}_C^\theta)$ is both $\mathcal{L}_{\text{alignment}}^\theta$ and $\mathcal{L}_{\text{uniformity}}^\theta$ are minimized.

$$\min(\mathcal{L}_C^\theta) \geq \min(\mathcal{L}_{\text{alignment}}^\theta) + \min(\mathcal{L}_{\text{uniformity}}^\theta).$$

Given an adequate encoder f^{θ^*}, that is flexible enough to minimize both $\mathcal{L}_{\text{alignment}}^\theta$ and $\mathcal{L}_{\text{uniformity}}^\theta$ for all transformation pairs ϕ_0 and ϕ_1, we will show we can construct another solution to the minimization problem that relies only on identity transformations. The alignment term is only minimized for all ϕ_0, ϕ_1, if $f^{\theta^*}\circ\phi_0(\boldsymbol{x}_i) = f^{\theta^*}\circ\phi_1(\boldsymbol{x}_i)$ for all $\boldsymbol{x}_i \sim \mathcal{M}$. So we know for f^{θ^*} that

$$f^{\theta^*} = \arg\min_{f^\theta} \mathcal{L}_{\text{alignment}}^\theta \quad \Longleftrightarrow \quad f^{\theta^*}\circ\phi_0(\boldsymbol{x}_i) = f^{\theta^*}\circ\phi_1(\boldsymbol{x}_i) \quad \forall\, \boldsymbol{x}_i \sim \mathcal{M}.$$

Define $\tilde{f} = f^{\theta^*}\circ\phi_0$. Since $f^{\theta^*}\circ\phi_0(\boldsymbol{x}_i) = f^{\theta^*}\circ\phi_1(\boldsymbol{x}_i)$,

$$\tilde{f}\circ\mathbb{I}(\boldsymbol{x}_i) = f^{\theta^*}\circ\phi_0(\boldsymbol{x}_i) = f^{\theta^*}\circ\phi_1(\boldsymbol{x}_i) \quad \forall\, \boldsymbol{x}_i \sim \mathcal{M}.$$

Using only the identity transformation $\mathbb{I}(x) = x$ for ϕ_0 and ϕ_1, and $\tilde{f}$ as the encoder in $\mathcal{L}_C$ yields the same minimal loss as under ϕ_0, ϕ_1 and f^{θ^*}.

□

A consequence of this result is that $\mathcal{L}_C^\theta$ with learnable transformations results in a data-independent loss function resulting in an anomaly score constant across samples. Plugging in $\phi_0(\boldsymbol{x}) = \phi_1(\boldsymbol{x}) = x$ cannot be used for anomaly detection as all samples (normal or not) achieve the same loss and anomaly score.

These propositions highlight a serious issue with using $\mathcal{L}_P^\theta$ or $\mathcal{L}_C^\theta$ for transformation learning and anomaly detection. Should the optimization reach the edge-cases of Propositions 1 and 2, $\mathcal{L}_P^\theta$ and $\mathcal{L}_C^\theta$ incur the same loss irrespective of whether the inputs are normal or abnormal data. Next, we prove that these edge-cases do not minimize DCL (Equation (2.2)).

2.2.1.2 Is DCL Suited for Transformation Learning?

Intuitively, the numerator of DCL (Equation (2.2)) encourages transformed embeddings to resemble their reference embedding (i.e., it encourages the semantics requirement), and the denominator encourages the diversity of transformations. The result of optimizing the DCL is a heterogeneous set of transformations that model various aspects of the data. We prove that the 'constant' edge-case and the 'identity' edge-case do not minimize DCL (Equation (2.2)).

Proposition 3. Let $\boldsymbol{c}_l \in \mathcal{Z}$ be a one-hot vector encoding the k^{th} position, and let $\sigma \in \mathbb{R}$ be a non-zero constant. The 'constant' edge-case $\mathcal{T}_k^\theta(\boldsymbol{x}) = \sigma \boldsymbol{c}_k$ for $k = 1, \dots, K$ does not minimize $\ell(\boldsymbol{x}; \theta)$ (Equation (2.2)) for any σ.

Proof. For simplicity, we define the embeddings obtained by the transformations as $\boldsymbol{z}_k := \mathcal{T}_k^\theta(\boldsymbol{x})$ for $k = 0, \dots, K$. We prove that the gradient of $\ell(\boldsymbol{x}; \theta)$ (Equation (2.2)) with respect to these embeddings evaluated at the 'constant' edge-case is nonzero, i.e., $\nabla \ell(\boldsymbol{x}; \theta) = [\frac{\partial \ell(\boldsymbol{x};\theta)}{\partial \boldsymbol{z}_0}, \dots, \frac{\ell(\boldsymbol{x};\theta)}{\partial \boldsymbol{z}_K}]^\top \neq 0$ at $\boldsymbol{z}_{1:K} = \sigma \boldsymbol{c}_{1:K}$, where $\boldsymbol{c}_k$ is a one-hot vector encoding the k^{th} position (i.e. $\boldsymbol{c}_{k,k} = 1$). Using the chain rule, the partial derivative $\frac{\partial \ell(\boldsymbol{x};\theta)}{\partial \boldsymbol{z}_u} \, \forall \, u \in \{0, \dots, K\}$ can be factorized as

$$\frac{\partial \ell(\boldsymbol{x}; \theta)}{\partial \boldsymbol{z}_u} = \sum_{k=1}^{K} \frac{\sum_{l \in \{1,\dots,K\}/\{k\}} h(\boldsymbol{z}_k, \boldsymbol{z}_l) \frac{\partial \left(\frac{\boldsymbol{z}_k^\top \boldsymbol{z}_l}{\|\boldsymbol{z}_k\| \|\boldsymbol{z}_l\|} - \frac{\boldsymbol{z}_k^\top \boldsymbol{z}_0}{\|\boldsymbol{z}_k\| \|\boldsymbol{z}_0\|} \right)}{\partial \boldsymbol{z}_u / \|\boldsymbol{z}_u\|}}{\sum_{l \in \{0,\dots,K\}/\{k\}} h(\boldsymbol{z}_k, \boldsymbol{z}_l)} \frac{I - \boldsymbol{z}_u \boldsymbol{z}_u^\top / \|\boldsymbol{z}_u\|^2}{\|\boldsymbol{z}_u\|}. \tag{2.7}$$

We plug in $\boldsymbol{z}_{1:K} = \sigma \boldsymbol{c}_{1:K}$ and then get

$$\left. \frac{\partial \ell(\boldsymbol{x}; \theta)}{\partial \boldsymbol{z}_u} \right|_{\boldsymbol{z}_{1:K} = \sigma \boldsymbol{c}_{1:K}} = \tag{2.8}$$

$$\sum_{k \in \{1,\dots,K\}/\{u\}} \left(\frac{\boldsymbol{c}_k^\top - \boldsymbol{z}_0^\top / \|\boldsymbol{z}_0\|}{h(\sigma \boldsymbol{c}_u, \boldsymbol{z}_0) + K - 1} + \frac{\boldsymbol{c}_k^\top}{h(\sigma \boldsymbol{c}_k, \boldsymbol{z}_0) + K - 1} \right) \frac{I - \boldsymbol{z}_u \boldsymbol{z}_u^\top / \|\boldsymbol{z}_u\|^2}{\|\boldsymbol{z}_u\|}.$$

The strategy of this proof is to show that the partial derivatives cannot all be zero. We will assume that the partial derivatives are zero and reach a contradiction through algebraic manipulation.

As σ is finite, by assuming the kth ($k \in \{1, \dots, K\}/\{u\}$) entry of $\frac{\partial \ell(\boldsymbol{x};\theta)}{\partial \boldsymbol{z}_u}$ equals zero at $\boldsymbol{z}_{1:K} = \sigma \boldsymbol{c}_{1:K}$ we get the kth entry of $\boldsymbol{z}_0$

$$\boldsymbol{z}_{0,k}^\top = \frac{\|\boldsymbol{z}_0\|}{K-1} \left(\boldsymbol{c}_{k,k}^\top + \boldsymbol{c}_{k,k}^\top \frac{h(\sigma \boldsymbol{c}_u, \boldsymbol{z}_0) + K - 1}{h(\sigma \boldsymbol{c}_k, \boldsymbol{z}_0) + K - 1} \right). \tag{2.9}$$

Similarly, by assuming kth entry of $\frac{\partial \ell(\boldsymbol{x};\theta)}{\partial \boldsymbol{z}_v}$ equals zero at $\boldsymbol{z}_{1:K} = \sigma \boldsymbol{c}_{1:K}$ with $v \neq u$ and $v \neq k$ we have

$$\boldsymbol{z}_{0,k}^\top = \frac{\|\boldsymbol{z}_0\|}{K-1} \left(\boldsymbol{c}_{k,k}^\top + \boldsymbol{c}_{k,k}^\top \frac{h(\sigma \boldsymbol{c}_v, \boldsymbol{z}_0) + K - 1}{h(\sigma \boldsymbol{c}_k, \boldsymbol{z}_0) + K - 1} \right). \tag{2.10}$$

As Equation (2.9) and Equation (2.10) are equal, we have $h(\sigma\boldsymbol{c}_u, \boldsymbol{z}_0) = h(\sigma\boldsymbol{c}_v, \boldsymbol{z}_0)$. Since this should hold for setting any of the entries of any of the partial derivatives, we get

$$h(\sigma\boldsymbol{c}_k, \boldsymbol{z}_0) = r, \quad \boldsymbol{z}_{0,k}^\top/\|\boldsymbol{z}_0\| = \frac{2}{K-1} \quad \forall\, k \in \{1, \ldots, K\}. \tag{2.11}$$

By plugging in $h(\sigma\boldsymbol{c}_k, \boldsymbol{z}_0) = r$ and $\boldsymbol{z}_{1:K} = \sigma\boldsymbol{c}_{1:K}$ to $\frac{\partial\ell(\boldsymbol{x};\theta)}{\partial\boldsymbol{z}_0}$ we have

$$\frac{\partial\ell(\boldsymbol{x};\theta)}{\partial\boldsymbol{z}_0}\Big|_{h(\sigma\boldsymbol{c}_k,\boldsymbol{z}_0)=r,\boldsymbol{z}_{1:K}=\sigma\boldsymbol{c}_{1:K}} = \frac{1-K}{r+K-1}\sum_{k=1}^{K}\boldsymbol{c}_k^\top\frac{I-\boldsymbol{z}_0\boldsymbol{z}_0^\top/\|\boldsymbol{z}_0\|^2}{\|\boldsymbol{z}_0\|}. \tag{2.12}$$

Equation (2.12) equals zero, if and only if every entry in the resulting vector equals zero. By assuming the kth ($k \in \{1, \ldots, K\}$) entry of $\frac{\partial\ell(\boldsymbol{x};\theta)}{\partial\boldsymbol{z}_0}$ equals zero at Equation (2.11) and $\boldsymbol{z}_{1:K} = \sigma\boldsymbol{c}_{1:K}$, we have

$$1 - K\left(\frac{2}{K-1}\right)^2 = 0\,, \tag{2.13}$$

which leads to a non-integral value of K. Since K is the number of transformations and is defined as an integer, we must conclude that Equation (2.12) cannot be zero. Therefore, $\nabla\ell(\boldsymbol{x};\theta) = [\frac{\partial\ell(\boldsymbol{x};\theta)}{\partial\boldsymbol{z}_0}, \ldots, \frac{\ell(\boldsymbol{x};\theta)}{\partial\boldsymbol{z}_K}]^\top \neq 0$ at $\boldsymbol{z}_{1:K} = \sigma\boldsymbol{c}_{1:K}$ and Proposition 3 is proven.

□

This result is expected as the DCL encourages the learned transformations to produce views that are predictive of the untransformed data. Specifically, the numerator of the DCL aims to align transformed views with the reference embedding of the same sample. The 'constant' edge-case violates this. The fact that the 'constant' edge-case is not a minimizer of the DCL makes DCL advantageous for transformation learning over the transformation prediction loss $\mathcal{L}_P^\theta$ (Proposition 1).

Next, we prove that the DCL also has an advantage over the contrastive learning loss $\mathcal{L}_C^\theta$. We show that (unlike $\mathcal{L}_C^\theta$, Proposition 2) the DCL is not minimized by identical transformations.

Proposition 4. The 'identity' edge-case $\forall_{k,l}\, \mathcal{T}_k^\theta(\boldsymbol{x}) = \mathcal{T}_l^\theta(\boldsymbol{x})$ does not minimize $\ell(\boldsymbol{x};\theta)$ (Equation (2.2)).

Proof. By plugging in $\mathcal{T}_{0:K}^\theta(\boldsymbol{x}) = \boldsymbol{z}$ to $\ell(\boldsymbol{x};\theta)$, we have

$$\ell(\boldsymbol{x};\theta)\big|_{\mathcal{T}_{0:K}^\theta(\boldsymbol{x})=\boldsymbol{z}} = -\sum_{k=1}^{K}\log\frac{h(\boldsymbol{z},\boldsymbol{z})}{Kh(\boldsymbol{z},\boldsymbol{z})} = K\log K. \tag{2.14}$$

It is sufficient to construct a set of transformations that have a smaller value of the loss. Assuming that the embedding space $\mathcal{Z}$ has enough dimensions, it is always possible to arrange them such that the first K transformations result in mutually orthogonal embeddings, while the last embedding is anti-aligned with the previous one:

$$\mathcal{T}_k^\theta(\boldsymbol{x}) \perp \mathcal{T}_l^\theta(\boldsymbol{x}) \quad \forall k, l = 0, ..., K-1 \text{ and } k \neq l, \tag{2.15}$$

$$\mathcal{T}_K^\theta(\boldsymbol{x}) = -\mathcal{T}_{K-1}^\theta(\boldsymbol{x}). \tag{2.16}$$

By plugging this set of transformations into the loss function $\ell(\boldsymbol{x};\theta)$ (Equation (2.2)) yields

$$\ell(\boldsymbol{x};\theta) = (K-1)\log K + \log(K - 1 + \exp(-1)) < K \log K. \tag{2.17}$$

Hence, the constructed example achieves a lower loss value than the 'identity' edge-case for any K. The 'identity' edge-case is therefore not minimizing $\ell(\boldsymbol{x};\theta)$. $\square$

This result shows that the 'identity' edge-case (or any other set of transformations that violates the diversity requirement) does not minimize the DCL, which makes the DCL superior to the contrastive learning loss $\mathcal{L}_C^\theta$ in learning transformations. The intuition behind this result is that the denominator of the DCL aims to push different transformations away from each other and assures that the transformed views are not identical.

Together Propositions 1 to 4 show that our proposed DCL is better suited for transformation learning than existing self-supervised losses that rely on transformations. However, in theory, we can still construct two possible failure cases of NTL in transformation learning or anomaly scoring.

2.2.2 Theoretical Limitations

NTL is not perfect. Even though NTL does not suffer from the failure cases in Propositions 3 and 4, it can, in theory, fail in learning transformations when the reference embeddings $\mathcal{T}_0^\theta(\boldsymbol{x})$ for all $\boldsymbol{x}$ collapse to the same constant, as shown in Proposition 5. We also theoretically show that it can fail in learning a data-dependent anomaly score even if the learned transformations are diverse and semantically meaningful in Proposition 6. To address these theoretical limitations, we provide practical recommendations on model implementation.

First, we construct a failure case for NTL in learning transformations when all the reference embeddings $\mathcal{T}_0^\theta(\boldsymbol{x})$ for all $\boldsymbol{x}$ collapse to the same constant, although we never observed this in practice. With constant reference embeddings, the model can end up with transformations that violate the semantics requirement.

Proposition 5. Assume the reference embeddings are collapsed to a constant on the unit hypersphere for all samples, namely, $\mathcal{T}_0^\theta(\boldsymbol{x})/\|\mathcal{T}_0^\theta(\boldsymbol{x})\| = \boldsymbol{c}$, where $\boldsymbol{c}$ is a constant vector. If there is any layer in the network of $\mathcal{T}_k^\theta$ with $k \in \{1, \ldots, K\}$ having a bias term $\boldsymbol{b}_k$, there exists an optimal constant solution, $\mathcal{T}_k^\theta(\boldsymbol{x}) = \boldsymbol{b}_k$, of $\mathcal{L}_{\text{NTL}}^\theta$ (Equation (2.2)).

Proof. Assume the parameters of $\mathcal{T}_k^\theta$ consist of the weights $\boldsymbol{w}_k$ and one bias term $\boldsymbol{b}_k$. By zeroing out the weights $\boldsymbol{w}_k$, we have $\forall\, \boldsymbol{x} \in \mathcal{X}\,, \mathcal{T}_k^\theta(\boldsymbol{x}) = \boldsymbol{b}_k$. Given a sample $\boldsymbol{x}$ and its reference embedding $\mathcal{T}_0^\theta(\boldsymbol{x})/\|\mathcal{T}_0^\theta(\boldsymbol{x})\| = \boldsymbol{c}$, we can find the optimal transformed embeddings $\boldsymbol{b}_k$ for $k = 1, \ldots, K$ by minimizing $\mathcal{L}_{\text{NTL}}^\theta$. Since by assumption $\forall\, \boldsymbol{x} \in \mathcal{X}\,, \mathcal{T}_0^\theta(\boldsymbol{x})/\|\mathcal{T}_0^\theta(\boldsymbol{x})\| = \boldsymbol{c}$, we have $\boldsymbol{b}_k$ for $k = 1, \ldots, K$ as the optimal transformed embeddings for all samples. Therefore, Proposition 5 is proven. $\square$

The assumption that the reference embeddings are collapsed to a constant can happen if the weights $\boldsymbol{w}_0$ in the network of $\mathcal{T}_0^\theta$ are all zero. Put differently, Proposition 5 implies that the collapse of the reference embedding should be prevented, and the bias terms should be removed in the networks of transformations since the networks can learn the constant functions to minimize $\mathcal{L}_{\text{NTL}}^\theta$ (Equation (2.2)).

We recommend the following parametrization of the transformations for preventing this in practice. We model $\mathcal{T}_0^\theta$ with an encoder $f^\theta(\cdot) : \mathcal{X} \to \mathcal{Z}$ which is parameterized by a network. The transformations $\mathcal{T}_k^\theta$ with $k \in \{1, \ldots, K\}$ are modeled by a composition of a bias-free network $\phi_k^\theta(\cdot) : \mathcal{X} \to \mathcal{X}$ and the encoder f^θ. The encoder f^θ serves as the shared feature extractor for all transformations. The weights in the network of f^θ cannot be all zero since a low DCL can only be achieved with distinguishable views. The data space transformation functions $\phi_k^\theta(\cdot)$ (for $k = 1, \ldots, K$) modeled by bias-free networks are optimized to be diverse and thus will not all collapse to constant solutions.

Second, we construct a failure case for NTL in learning a data-dependent anomaly score in theory, even if the learned transformations are diverse and semantically meaningful. The success of the scoring of self-supervised anomaly detection methods is based on the concept of “inlier priority” [40], where the model is generalized better to unseen normal data than anomalies as the majority of the training data is normal. At test time, the model is expected to perform differently on normal and abnormal data. We argue that a generalization gap is required in self-supervised anomaly detection.

Requirement 3 (Generalization Gap)**.** The model should have a generalization gap between unseen inliers and outliers in solving the auxiliary task; namely, it performs on inliers and outliers differently.

Here we show a failure case of NTL in anomaly scoring even though the learned transformations are diverse and semantically meaningful.

Proposition 6. Using any rotation matrices as the transformations $\mathcal{T}_k^\theta(\boldsymbol{x}) = R_k\boldsymbol{x}$ for $k = 1, \ldots, K$ and using the identity transformation as $\mathcal{T}_0^\theta(\boldsymbol{x}) = \boldsymbol{x}$ leads to a constant data-independent score function (Equation (2.4)).

Proof. R_k rotates the vector by an angle ρ_k counterclockwise. Plugging the transformed embeddings $R_k\boldsymbol{x}$ and the reference embedding $\boldsymbol{x}$ in the anomaly score function (Equation (2.4)) leads to

$$s(\boldsymbol{x}) = -\sum_{k=1}^{K} \log \frac{\exp(\cos \rho_k)}{\exp(\cos \rho_k) + \sum_{l \in \{1,\ldots,K\}/\{k\}} \exp(\cos \rho_k - \rho_l)},$$

which is a constant function independent of the input sample $\boldsymbol{x}$. □

Proposition 6 implies that parameterizing the neural transformations as linear affine transformations has the risk of leading to a constant score function independent of the input samples. A practical solution to avoid this is to learn non-linear transformations by including non-linear activations (e.g., ReLU activation) in the network of transformations.

2.3 Applications of NTL

NTL is applicable to a variety of domains: it can be used to detect abnormal time series, and it is applicable to anomaly detection on tabular data. The advantages of NTL include that it outperforms other deep anomaly detection methods (as our experimental results in this section will show) and that it does not require hand-crafted data augmentation schemes for specialized domains. Since the neural transformations are *learned* together with the other architecture components, they automatically lead to an appropriate data augmentation scheme. In fact, since the transformations do not need to be hand-crafted, they can even be applied to data representations that are not easily accessible to human intuition, such as image features that are extracted from an image using a pre-trained neural network, or to word embeddings from a language model. This allows us to use NTL to also perform anomaly detection on images and on text.

In this section, we present the following applications of NTL: (i) Section 2.3.2: Anomaly detection on time series. We evaluate NTL on identifying whole abnormal sequences. We find that it can effectively find anomalies in audio signals, health care signals, as well as motion signals. (ii) Section 2.3.3: Anomaly detection on tabular data. We find that NTL is the most effective method for detecting anomalies in four real-world datasets from the medical and cyber security domains. (iii) Section 2.3.4: Anomaly detection on image data. We study NTL on raw images and image features. (iv) Section 2.3.5: Anomaly segmentation on image data. We study NTL on localizing anomalous regions of an image. (v) Section 2.3.6: Anomaly detection on text. We build NTL on word embeddings to detect whole abnormal sentences. For each application, we specify the choice of neural transformations $\mathcal{T}_k^\theta$. We present an extensive empirical evaluation in comparison with other successful approaches for anomaly detection and discuss the results and their implications. We also provide an analysis of NTL on synthetic data to visualize the learned transformations and their contributions to the anomaly score in Appendix A.1 and a sensitivity study of transformation design choices in Appendix A.2.

2.3.1 Evaluation Protocol

An anomaly detection method is useful if it is good at detecting anomalies in unseen test data. To test this, we follow the standard evaluation protocol for anomaly detection in the literature [e.g., 22, 28, 37, 38, 39, 41, 42, 89, 95, 96, 97, 98, 99]. Since it is hard to find labeled test sets for anomaly detection in many domains, a common approach for setting up an evaluation pipeline is to repurpose classification datasets. We consider two such evaluation protocols: the standard 'one-vs-rest' and the more challenging 'n-vs-rest' evaluation protocol. Both settings turn a classification dataset into a quantifiable anomaly detection benchmark.

One-vs-rest. This evaluation setup has been used in virtually all recent papers on deep anomaly detection published at top-tier venues [e.g., 22, 28, 37, 38, 39, 41, 42, 89, 95, 96, 97, 98, 99]. For 'one-vs.-rest', the dataset is split by the $\mathcal{N}$ class labels, creating $\mathcal{N}$ one-class classification task, in each of which one class is considered normal, and only samples drawn from this class are used as training data. All other classes are considered abnormal. The test and validation sets contain samples from all classes, including the normal class. The samples from the other classes should be detected as anomalies. By iterating over the classes and changing which class is considered normal, we obtain $\mathcal{N}$ separate anomaly detection tasks for evaluation.

N-vs-rest. An alternative evaluation protocol we also consider follows the insight that it is more realistic in practice that the "normal distribution" is more diverse than a single class (i.e., it is more realistic that it contains multiple classes [100, 101]). So we also evaluate methods on the more challenging n-vs.-rest protocol, where n classes (for $1 < n < \mathcal{N}$) are treated as normal, and the remaining classes provide the anomalies in the test and validation set. By increasing the variability of what is considered normal data, anomaly detection becomes more challenging.

2.3.2 Anomaly Detection on Time Series

Our goal is to detect abnormal time series on a *whole-sequence level.* This is a different set-up than anomaly detection *within* time series but equally important in practice. For example, one might be interested in detecting abnormal sound or finding production quality issues by detecting abnormal sensor measurements recorded over the duration of producing a batch. Other applications include sports and health monitoring; e.g., finding abnormal movement patterns during sports can be indicative of fatigue or injury.

In this section, we first present the datasets and baselines we use to study NTL. We then present the implementation details and finally describe the empirical results. We find that NTL learns semantically meaningful and diverse transformations and detects anomalous time series successfully.

2.3.2.1 Datasets and Baselines

Time series datasets. We select datasets from various domains. The datasets come from the UEA multivariate time series classification archive[1] [103]. We evaluate NTL on them with both one-vs-rest setting and n-vs-rest setting.

- Spoken Arabic Digits (SAD): Sound of ten Arabic digits, spoken by 88 speakers. The samples are stored as 13 Mel Frequency Cepstral Coefficients. We select sequences with lengths between 20 and 50 and get a dataset of 7824 samples. The sequences that are shorter than 50 are zero-padded to have a length of 50.
- Naval Air Training and Operating Procedures Standardization (NATOPS): The data is from a motion detection competition of various movement patterns used to control planes in naval air training. The data has six classes of distinct actions. The dataset has 360 samples, each being a sequence of x, y, z coordinates for eight body parts of length 51.
- Character Trajectories (CT): The data consists of 2858 character samples from 20 classes. Each instance is a 3-dimensional pen tip velocity trajectory. The data is truncated to the length of the shortest, which is 182.
- Epilepsy (EPSY): The data was generated with healthy participants simulating four different activities: walking, running, sawing with a saw, and seizure mimicking whilst seated. The dataset has 275 samples, each being a 3-dimensional sequence of length 203.
- Racket Sports (RS): The data is a record of university students playing badminton or squash. The data records the x, y, z coordinates for both the gyroscope and accelerometer. Sport and stroke types separate the data into four classes. The dataset has 303 samples, each being a 6-dimensional sequence with a length of 30.

[1]We selected datasets on which supervised multi-class classification methods achieve strong results [102]. Only datasets with separable classes can be repurposed for anomaly detection.

Time series baselines. We study NTL in comparison to unsupervised and self-supervised anomaly detection methods. They include three anomaly detection baselines: OCSVM [13], IF [15], a tree-based model which aims to isolate anomalies, and LOF [14], which uses density estimation with K-Nearest Neighbor (KNN).

Next, we include two deep anomaly detection methods, Deep Support Vector Data Description (DSVDD) [89], which fits a one-class classification in the feature space of a neural net, and Deep Autoencoding Gaussian Mixture Model (DAGMM) [104], which estimates the density in the latent space of an autoencoder. We also include two baselines that are specifically designed for time series data: RNN-based Model (RNN) directly models the data distribution $p(\boldsymbol{x}_{1:T}) = \prod p(\boldsymbol{x}_t|\boldsymbol{x}_{<t})$ and uses the log-likelihood as the anomaly score. LSTM-based Encoder-Decoder (LSTM-ED) [105] is an encoder-decoder time series model where the anomaly score is based on the reconstruction error.

Finally, We choose two self-supervised baselines, which are technically also deep anomaly detection methods. GOAD [41] is a distance-based classification method based on random affine transformations. Golan and El-Yaniv [38] is a softmax-based classification method based on hand-crafted transformations, which show impressive performance on images. We adopt their pipeline to time series here by crafting specific time series transformations (fixed Ts). Their implementation details are provided in Appendix B.1.

2.3.2.2 NTL on Time Series

We consider a dataset of time series $\boldsymbol{x} \in \mathbb{R}^{d\times T}$, where d is the data dimension and T is the time series length. $\mathcal{T}_0^\theta$ is modeled by an encoder f^θ. The transformations $\mathcal{T}_k^\theta$ with $k \in \{1,\ldots,K\}$ are modeled by composing data space transformation functions ϕ_k^θ with the shared encoder f^θ. We consider two parametrizations of the transformation function ϕ_k^θ: *residual* $\phi_k^\theta(\boldsymbol{x}) := M_k^\theta(\boldsymbol{x}) + \boldsymbol{x}$ with $M_k^\theta(\boldsymbol{x}) \in \mathbb{R}^d$, and *multiplicative* $\phi_k^\theta(\boldsymbol{x}) := M_k^\theta(\boldsymbol{x}) \odot \boldsymbol{x}$ with $M_k^\theta(\boldsymbol{x}) \in (0,1)^d$. Both M_k^θ and the encoder f^θ are modeled by convolutional neural networks.

Implementation details. M_k^θ is a neural network that consists of one 1D convolutional layer on the bottom, a stack of three residual blocks of 1D convolutional layers with affine-free instance normalization layers and ReLU activations, as well as one 1D convolutional layer on the top. All bias terms in the network are fixed as zero. The network of the encoder f^θ consists of several residual blocks of 1D convolutional layers with a hidden dimension of 32, as well as one final linear layer to extract the embeddings. The number of residual blocks depends on the data dimension. Specifically, we use four residual blocks in the encoder for RS, five residual blocks for SAD and NATOPS, seven residual blocks for CT and EPSY. The output dimensions

Table 2.2: Average AUC (%) with standard deviation for one-vs-rest anomaly detection on time series datasets. NTL achieves the best results on 4 of 5 datasets.

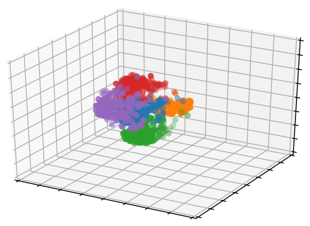
	SAD	NATOPS	CT	EPSY	RS	Avg.
OCSVM	95.3	86.0	97.4	61.1	70.0	82.0
IF	88.2	85.4	94.3	67.7	69.3	81.0
LOF	98.3	89.2	97.8	56.1	57.4	79.8
RNN	81.5±0.4	89.5±0.4	96.3±0.2	80.4±1.8	84.7±0.7	86.5
LSTM-ED	93.1±0.5	91.5±0.3	79.0±1.1	82.6±1.7	65.4 ±2.1	82.3
DSVDD	86.0±0.1	88.6±0.8	95.7±0.5	57.6±0.7	77.4±0.7	81.1
DAGMM	80.9±1.2	78.9±3.2	89.8±0.7	72.2±1.6	51.0±4.2	74.6
GOAD	94.7±0.1	87.1±1.1	97.7±0.1	76.7±0.4	79.9±0.6	87.2
fixed Ts	96.7±0.1	78.4±0.4	97.9±0.1	80.4±2.2	**87.7±0.8**	88.2
NTL	**98.9±0.1**	**94.5±0.8**	**99.3±0.1**	**92.6±1.7**	86.5±0.6	**94.4**

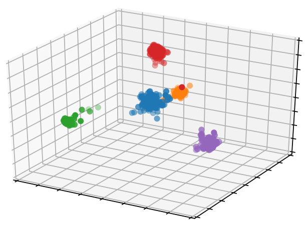
of the encoders are 32 for SAD, 128 for EPSY, and 64 for others. On all time series datasets, we set the number of transformations $K = 11$ and the temperature $\tau = 0.1$. More implementation details are provided in Appendix B.1.

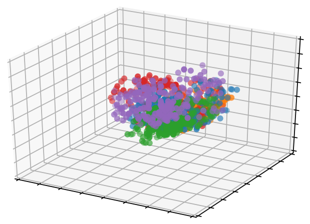
2.3.2.3 Empirical Results

The results of NTL in comparison to the baselines on time series datasets from various fields are reported in Table 2.2. NTL improves the detection accuracy over existing baselines in terms of AUC by 7.2% on average. NTL outperforms all shallow baselines in all experiments and outperforms the deep learning baselines in 4 out of 5 experiments. Only on the RS data, it is outperformed by transformation prediction with fixed transformations, which we designed to understand the value of learning transformations with NTL vs. using hand-crafted transformations. The results confirm that designing the transformations only succeeds sometimes, whereas with NTL we can learn the appropriate transformations. The learned transformations also give NTL a competitive advantage over the other self-supervised baseline GOAD, which uses random affine transformations. The performance of the traditional anomaly detection baselines hints at the difficulty of each anomaly detection task; the traditional methods perform well on SAD and CT, but perform worse than the deep learning-based methods on other data.

What does NTL learn? For visualization purposes, we train NTL with the learnable transformations on the SAD data. Figure 2.2 shows the structure in the data space $\mathcal{X}$ and the embedding space of the encoder $\mathcal{Z}$ after training. Held-out data samples (blue) are transformed by each of the learned transformations with the multiplicative transformation function $\phi_k^\theta(\boldsymbol{x}) = M_k^\theta(\boldsymbol{x}) \odot \boldsymbol{x}$ to produce $K = 4$ different views of


(a) normal data on $\mathcal{X}$


(b) normal data on $\mathcal{Z}$


(c) anomalies on $\mathcal{X}$

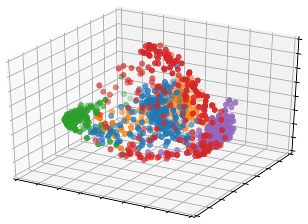

(d) anomalies on $\mathcal{Z}$

Figure 2.2: 3D visualization (projected using PCA) of how the original samples (blue) from the SAD dataset and the different views created by the neural transformations of NTL (one color per transformation type) cluster in data space (Figures 2.2a and 2.2c) and in the embedding space of the encoder (Figures 2.2b and 2.2d). The crisp separation of the different transformations of held-out inliers (Figure 2.2b) in contrast to the overlap between transformed anomalies (Figure 2.2d) visualizes how NTL is able to detect anomalies.

each sample (the transformations are color-coded by the other colors). Projection to three principal components with PCA allows for visualization in 3D. In Figures 2.2a and 2.2c, we can see that the transformations already cluster together in the data space, but with the help of the encoder, the different views of inliers are separated from each other (Figure 2.2b). In comparison, the anomalies and their transformations are less structured in $\mathcal{Z}$ (Figure 2.2d), visually explaining why they incur a higher anomaly score and can be detected as anomalies.

The learned masks $M^{\theta}_{1:4}(\boldsymbol{x})$ of one inlier $\boldsymbol{x}$ are visualized in Figure 2.3. We can see that the four masks are dissimilar from each other and have learned to focus on different aspects of the spectrogram. The masks take values between 0 and 1, with dark areas corresponding to values close to 0 that are zeroed out by the masks, while light colors correspond to the areas of the spectrogram that are not masked out. Interestingly, in M^{θ}_1, M^{θ}_2, and M^{θ}_3, we can see 'black lines' where they mask out entire frequency bands at least for part of the sequence. In contrast, M^{θ}_4 has a bright spot in the middle left part of the spectrogram; it creates views that focus on the content of the intermediate frequencies in the first half of the recording.

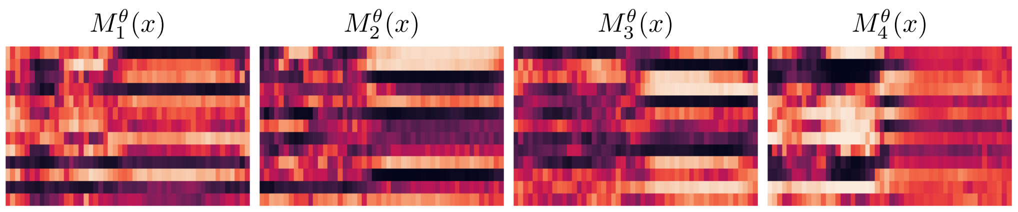
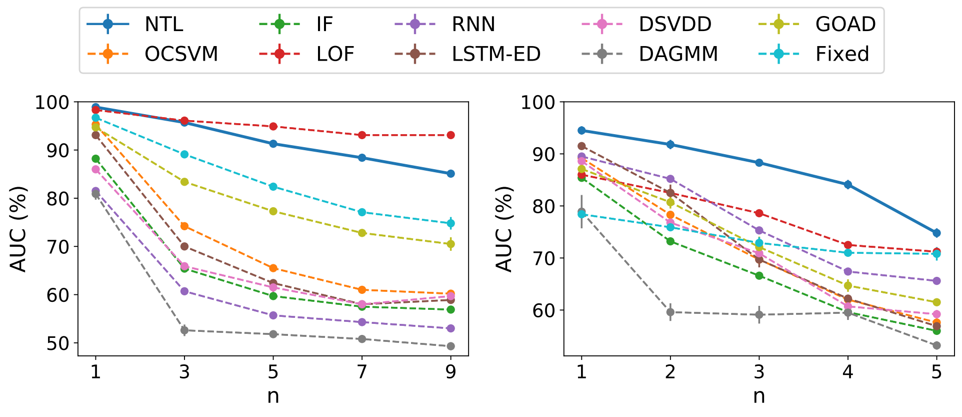


Figure 2.3: NTL learns dissimilar masks for SAD spectrograms (whose x-axis represents time and the y-axis represents frequency). Dark horizontal lines indicate where M_1^θ and M_2^θ mask out frequency bands almost entirely, while the bright spot in the middle left part of M_4^θ indicates that this mask brings the intermediate frequencies in the first half of the recording into focus.



(a) AUCs on SAD (b) AUCs on NATOPS

Figure 2.4: AUC results of n-vs-all experiments on SAD and NATOPS for varying n. NTL outperforms all baselines on NATOPS and all deep learning baselines on SAD. LOF, a method based on KNN, outperforms NTL, when $n > 3$ on SAD.

How do the methods cope with increased variability of inliers? To study this question empirically, we increase the number of classes n considered to be inliers. We test all methods on SAD and NATOPS under the n-vs-rest setting with varying n. Since there are too many combinations of normal classes when n approaches $\mathcal{N} - 1$, we only consider combinations of n consecutive classes. From Figure 2.4 we can observe that the performance of all methods drops as the number of classes included in the normal data increases. This shows that the increased variance in the nominal data makes the task more challenging. NTL outperforms all baselines on NATOPS and all deep-learning baselines on SAD. It is interesting that LOF, a method based on KNN, performs better than our method (and all other baselines) on SAD when n is larger than three.

Table 2.3: Average AUC (%) with standard deviation for n-vs-rest ($n = \mathcal{N} - 1$) anomaly detection on time series datasets. NTL outperforms all deep methods and achieves the best performance on average.

	SAD	NATOPS	CT	EPSY	RS	Avg.
OCSVM	60.2	57.6	57.8	50.2	55.9	56.3
IF	56.9	56.0	57.9	55.3	58.4	56.9
LOF	**93.1**	71.2	**90.3**	54.7	59.4	73.7
RNN	53.0±0.1	65.6±0.4	55.7±0.8	74.9±1.5	75.8±0.9	64.9
LSTM-ED	58.9±0.5	56.9±0.7	50.9±1.2	56.8±2.1	63.1±0.6	57.3
DSVDD	59.7±0.5	59.2±0.8	54.4±0.7	52.9±1.4	62.2±2.1	57.7
DAGMM	49.3±0.8	53.2±0.8	47.5±2.5	52.0±1.0	47.8±3.5	50.0
GOAD	70.5±1.4	61.5±0.7	81.1±0.1	62.7±0.9	68.2±0.9	68.8
fixed Ts	74.8±1.3	70.8±1.3	63.0±0.6	69.8±1.6	**81.6±1.2**	72.0
NTL	85.1±0.3	**74.8±0.9**	87.4±0.2	**80.5±1.0**	80.0±0.4	**81.6**

We also include quantitative results for $n = \mathcal{N} - 1$ under the n-vs-rest setting for all time series datasets, where only one class is considered abnormal, and the remaining $\mathcal{N} - 1$ classes are normal. The results are reported in Table 2.3. NTL raises the detection accuracy in terms of AUC by 7.9% on average and outperforms other deep learning methods on 4 out of 5 datasets. On RS, it is just outperformed by transformation prediction with hand-crafted transformations. The traditional method LOF performs better than deep learning methods on CT and SAD.

2.3.3 Anomaly Detection on Tabular Data

Tabular data is another important application area of anomaly detection. For example, many types of health data come in tabular form. To unleash the power of self-supervised anomaly detection for these domains, Bergman and Hoshen [41] suggest using random affine transformation. Here we study the benefit of *learning* the transformations on tabular data with NTL on four datasets, Arrhythmia, Thyroid, KDD, and KDDRev.

2.3.3.1 Datasets and Baselines

Tabular datasets. We base our empirical study of tabular anomaly detection on previous work [41, 104] and follow their choice of datasets as well as their precedent of reporting results in terms of F1-scores. The datasets include the small-scale medical datasets Arrhythmia and Thyroid as well as the large-scale cyber intrusion detection datasets KDD and KDDRev. We follow the configuration of Bergman and Hoshen [41] to train all models on half of the normal data and test on the rest of the normal data as well as the anomalies.

- Arrhythmia: A cardiology dataset from the UCI repository contains 274 continuous attributes and five categorical attributes.
- Thyroid: A medical dataset from the UCI repository contains attributes related to hyperthyroid diagnosis.
- KDD: The KDDCUP99 10 percent dataset from the UCI repository contains 34 continuous and seven categorical attributes.
- KDDRev: It is derived from the KDDCUP99 10 percent dataset. The non-attack samples are considered normal, and the attack samples are considered abnormal.

On all datasets, we use the same preprocessing steps as Bergman and Hoshen [41].

Tabular baselines. We compare NTL to shallow anomaly detection baselines, including OCSVM [13], IF [15], and LOF [14], and to deep anomaly detection methods DSVDD [89], DAGMM [104], GOAD [41], and Deep Robust One-Class Classification (DROCC) [81]. The results of OCSVM, LOF, DAGMM, and GOAD are from Bergman and Hoshen [41]. We obtained the results for DSVDD and DROCC using the implementation provided with the respective publications.

2.3.3.2 NTL on Tabular Data

We consider a dataset of tabular data $\boldsymbol{x} \in \mathbb{R}^d$. $\mathcal{T}_0^\theta$ is modeled by an encoder f^θ. The transformations $\mathcal{T}_k^\theta$ with $k \in \{1, \ldots, K\}$ are modeled by composing data space transformation functions ϕ_k^θ with the shared encoder f^θ. We consider two parametrizations of the transformation function ϕ_k^θ: *residual* $\phi_k^\theta(\boldsymbol{x}) := M_k^\theta(\boldsymbol{x}) + \boldsymbol{x}$ with $M_k^\theta(\boldsymbol{x}) \in \mathbb{R}^d$, and *multiplicative* $\phi_k^\theta(\boldsymbol{x}) := M_k^\theta(\boldsymbol{x}) \odot \boldsymbol{x}$ with $M_k^\theta(\boldsymbol{x}) \in (0,1)^d$. M_k^θ and the encoder f^θ are both modeled by feed-forward neural networks.

Implementation details. The neural network of M_k^θ consists of two bias-free linear layers with an intermediate ReLU activation. When using the multiplicative parametrization, it has a sigmoid activation on the final layer. We use the residual parametrization for the neural transformations on Thyroid and Arrhythmia, and use the multiplicative parametrization on KDDCUP, and KDDCUP-Rev. The neural network of the encoder f^θ consists of five bias-free linear layers with ReLU activations. The output dimensions of the encoder are 24 for Thyroid and 32 for the other datasets. We set the number of neural transformations $K = 11$ and the temperature $\tau = 0.1$ on all tabular datasets.

2.3.3.3 Empirical Results

In line with prior work, we use the average F1-score (%) with standard deviation as the evaluation metric and report the results in Table 2.4. The results of OCSVM, LOF, DAGMM, and GOAD are taken from Bergman and Hoshen [41]. For DSVDD

Table 2.4: F1-score (%) with standard deviation for anomaly detection on tabular datasets (choice of F1-score consistent with prior work). NTL achieves the best results on all datasets.

	Arrhythmia	Thyroid	KDD	KDDRev	Avg.
OCSVM	45.8	38.9	79.5	83.2	61.9
IF	57.4	46.9	90.7	90.6	71.4
LOF	50.0	52.7	83.8	81.6	67.0
DSVDD	53.9±3.1	70.8±1.8	99.0±0.1	98.6±0.2	80.6
DAGMM	49.8	47.8	93.7	93.8	71.3
GOAD	52.0±2.3	74.5±1.1	98.4±0.2	98.9±0.3	81.0
DROCC	47.5±0.7	75.6±1.8	92.9±0.4	95.2±1.0	77.8
NTL	**60.3±1.1**	**76.8±1.9**	**99.3±0.1**	**99.1±0.1**	**83.9**

and DROCC, we run their official implementations and report the results. NTL outperforms all baselines on all tabular datasets. In particular, NTL raises the F1-score on Arrhythmia by 3.7% and on Thyroid by 2.4%. Compared with the self-supervised baseline GOAD with random affine transformations, the neural transformations learned from data lead to better detection accuracy. In addition, we find that NTL needs fewer transformations than GOAD. On the medical datasets, for example, GOAD uses 256 transformations, while NTL achieves superior performance with only 11 transformations.

2.3.4 Anomaly Detection on Image Data

Anomaly detection on images is important for many applications and has received a lot of attention in the last few years. Recent works [29, 30, 37] employ shallow anomaly detectors (KNN or OCSVM) upon the image features obtained from a pre-trained model, often achieving better detection accuracy than self-supervised anomaly detection on raw images. This raises the question of how to apply self-supervised anomaly detection to image features which might lead to an additional performance boost. However, applying self-supervised anomaly detection methods (such as Golan and El-Yaniv [38], Wang et al. [40], Bergman and Hoshen [41]) to image features is difficult since all of them rely on hand-crafted data augmentations designed for raw images. Designing transformations manually requires intuitions about image invariances (e.g., humans still recognize dogs in rotated images of dogs). These intuitions do not readily generalize to image features. In contrast, NTL learns transformations and hence requires no intuition. For this reason, it can be directly used with image features.

In this section, we first present the datasets and baselines used to study NTL. We then present how to apply neural transformations to raw images and image features.

We finally show that when applying NTL on image features, we learn semantically meaningful and diverse transformations and achieve competitive detection accuracy.

2.3.4.1 Datasets and Baselines

Image datasets. We study three image datasets, FashionMINST, CIFAR10, and CIFAR100, which are commonly used in previous works [e.g., 29, 30, 37, 38, 89]. On FMNIST, we study the one-vs-rest setting. On the more challenging datasets, CIFAR10 and CIFAR100 [2], we consider both the one-vs-rest setting and the n-vs-rest setting with $n = \mathcal{N} - 1$ (the number of classes $\mathcal{N}$).

Image baselines. We compare NTL with modern baselines based on different techniques. We include one deep anomaly detection baseline, DSVDD [89], which learns a compact boundary of normal data in the embedding space to detect anomalies. We also include Geometric Transformation Classification (GEOM) [38] as the self-supervised baseline, which proposes an auxiliary transformation classification task for anomaly detection. Next, we include three baselines, which build anomaly detectors upon image features. They are Distribution-Augmented Contrastive Learning (Contra-DA) [37], which applies shallow anomaly detectors (e.g., OCSVM) on features from an encoder pre-trained with a contrastive loss, Deep Nearest-Neighbors (DN2) [29], which uses a KNN-based detector on features obtained from a pre-trained ResNet, and Pretrained Anomaly Detection Adaptation (PANDA) [30], the state-of-the-art method improves over DN2 by fine-tuning the pre-trained ResNet.

2.3.4.2 NTL for Image Anomaly Detection

NTL learns data-dependent transformations automatically. Therefore, we can apply neural transformations to raw images or semantic features obtained from existing pre-trained feature extractors. For both variants, we set the temperature $\tau = 0.1$. $\mathcal{T}_0^\theta$ is modeled by an encoder f^θ. The transformations $\mathcal{T}_k^\theta$ with $k \in \{1, \ldots, K\}$ are modeled by composing transformation functions ϕ_k^θ with the shared encoder f^θ.

NTL on raw images (NTL-I). We consider a dataset of images $\boldsymbol{x}$. The data space transformation function is parametrized multiplicatively as $\phi_k^\theta(\boldsymbol{x}) = \boldsymbol{x} \odot M_k^\theta(\boldsymbol{x})$, where $M_k^\theta(\boldsymbol{x})$ is an attention mask (with values between 0 and 1) and the multiplication is applied element-wise. M_k^θ is modeled by a residual convolutional network with affine-free batch normalization layers and a sigmoid activation on the top. All bias terms in the network are fixed as zero. We set the number of neural transformations $K = 15$ for all image datasets. As in Bergman and Hoshen [41], we use a ResNet architecture as the encoder f^θ with the output dimension of 64.

[2]For CIFAR100, we use 20 super-class labels.

Table 2.5: Average AUC (%) with standard deviation on image datasets. NTL-F performs comparably to the state-of-the-art method, PANDA, which relies on the fine-tuning of the large pre-trained feature extractor and a costly detection.

Datasets	DSVDD	GEOM	Contra-DA	DN2	PANDA	NTL-I	NTL-F
FMNIST	92.8	93.5	94.8±0.3	94.4	**95.6**	94.5±0.1	94.9±0.04
CIFAR10	64.8	86.0	92.5±0.6	92.5	**96.2**	69.6±0.3	95.3±0.06
CIFAR100	-	78.7	86.5±0.7	89.3	**94.1**	68.1±0.1	93.2±0.12
Avg. rank	7.0	5.3	3.3	3.7	1.0	5.3	2.0

NTL on image features (NTL-F). We process images with a pre-trained ResNet and obtain a dataset of image features $\boldsymbol{x} \in \mathbb{R}^{2048}$ as in Bergman et al. [29]. We consider two parametrizations of the transformation function: *forward* $\phi_k^\theta(\boldsymbol{x}) := M_k^\theta(\boldsymbol{x})$ with $M_k^\theta(\boldsymbol{x}) \in \mathbb{R}^{2048}$, and *multiplicative* $\phi_k^\theta(\boldsymbol{x}) := M_k^\theta(\boldsymbol{x}) \odot \boldsymbol{x}$ with $M_k^\theta(\boldsymbol{x}) \in (0,1)^{2048}$. $M_k^\theta(\boldsymbol{x})$ is computed by three bias-free linear layers with affine-free batch normalization layers and ReLU activations. A sigmoid activation is applied on the final layer of M_k^θ when using the multiplicative parametrization. We set the number of neural transformations $K = 15$. The encoder f^θ is modeled by two linear layers with an intermediate ReLU activation. The output dimension of the encoder is 256.

2.3.4.3 Empirical Results

The results of NTL in comparison to the baselines on image datasets with the one-vs-rest setting are reported in Table 2.5, where the results of baselines are taken from Reiss et al. [30], Sohn et al. [37], Golan and El-Yaniv [38], Ruff et al. [89]. When neural transformations are applied to raw images directly, NTL-I performs better than DSVDD but worse than GEOM using hand-crafted transformations. It turns out that learning good transformations from noisy raw images is still hard, and the learned transformations are not competitive with transformations selected by human experts on raw images. One reason for the poor performance on raw images could be that the learned transformations affect mostly low-level features and do not interact with higher-level semantic features, as would be desired for a strong self-supervision task. This motivates applying NTL to pre-trained image features.

The detection results of NTL-F on image features (obtained with a ResNet pre-trained on ImageNet) are reported in Table 2.5. NTL-F outperforms DN2 using the same pre-trained feature extractor, and all baselines with no access to a feature extractor are pre-trained on ImageNet. PANDA outperforms NTL-F but also requires a careful fine-tuning of the large pre-trained ResNet and has a risk of performance degradation caused by catastrophic forgetting [30]. Moreover, using PANDA at test time is expensive; the KNN-based detection is slower than the forward-pass of the other methods in Table 2.5, and it requires memorizing the entire training data for

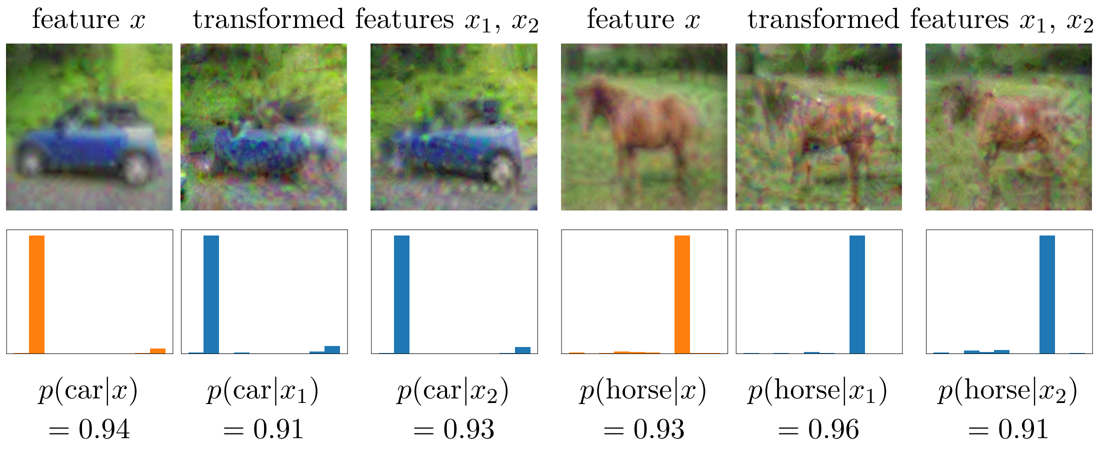


Figure 2.5: Visualization and semantics checking examples from CIFAR10. By inverting the original and transformed image features, we visualize the features in the image space in the first row. In the second row, we plot their class prediction results from a downstream classifier. The transformed features are diverse and still preserve semantic information.

test time. Our method achieves the highest performance among all methods that do not suffer from this drawback. When using PANDA with a more efficient distance for detection, the results will be 2% lower (e.g., CIFAR10: from 96.2% to 94.2%) [30], which is outperformed by NTL-F (95.3%). It is even hard for human experts to design good transformations on the feature vectors, while NTL can learn them from data automatically. NTL offers a convenient way to perform self-supervised anomaly detection on the image features to utilize the powerful existing pre-trained models continuously being developed and improved by the vision community.

What do the transformed features look like? For an interpretable visualization of the transformed features learned by NTL-F, we follow Mahendran and Vedaldi [106] to invert them back to the image space by seeking an image that best matches the source representation. From the resulting images shown in the first row of Figure 2.5, we can see that the transformations disrupt different local regions and textures of the image but preserve the global shape of the object. The learned local disruptions preserve global semantics and vary, thereby satisfying the diversity requirement.

Furthermore, we analyze the semantic information contained in the transformed features by checking the class prediction of a downstream classifier[3] trained on the raw features and ground truth class labels. In the second row of Figure 2.5 we can see that the predictions given transformed features (blue) perfectly match the

[3]The classifier consists of two linear layers of units [128,10] with an intermediate ReLU activation.

predictions given the raw features (orange)[4]. This confirms the claim that the learned transformations manipulate the image without changing the global semantics as encoded by the class label.

How do the methods cope with increased variability of inliers? To study how sensitive NTL-F is to the variability of inliers, we also consider the n-vs-rest setting with $n = \mathcal{N} - 1$. As shown in Table 2.6, NTL-F outperforms baselines on CIFAR10 but performs worse than DN2 on CIFAR100. This is consistent with the results in time series anomaly detection in Table 2.3, where the KNN-based methods (DN2 in vision and LOF in sequences) have an advantage in anomaly detection when the normal distribution contains multiple modes.

Table 2.6: Average AUC (%) with standard deviation on CIFAR10 and CIFAR100 under n-vs-rest setting with $n = \mathcal{N} - 1$. NTL outperforms the baselines on average.

Datasets	GEOM	DN2	NTL-F
CIFAR10	61.7	71.7	**77.7±0.4**
CIFAR100	57.3	**71.0**	68.0±0.2

2.3.5 Anomaly Segmentation on Image Data

Anomaly detection on images predicts one anomaly score for each image. Anomaly segmentation on images is a more complex problem that requires pixel-level predictions. The predicted anomaly heatmap provides reasonable explanations of why the model detects an image anomalous.

In this section, we first present the dataset and baselines used to study NTL for anomaly segmentation. We then present how to apply neural transformations for segmenting anomalous image regions. We finally show the quantitative and qualitative results of NTL on anomaly segmentation.

2.3.5.1 Datasets and Baselines

Image datasets. We evaluate the anomaly segmentation performance of NTL on MVTEC. MVTEC [107] is an industrial anomaly segmentation dataset. It contains images from ten object classes and five texture classes. The abnormal images have pixel-level annotated defective regions. Many recent works [88, 108, 109, 110, 111] build an anomaly segmentation benchmark on MVTEC.

[4]Note that the transformed features have not been used to train the classifier

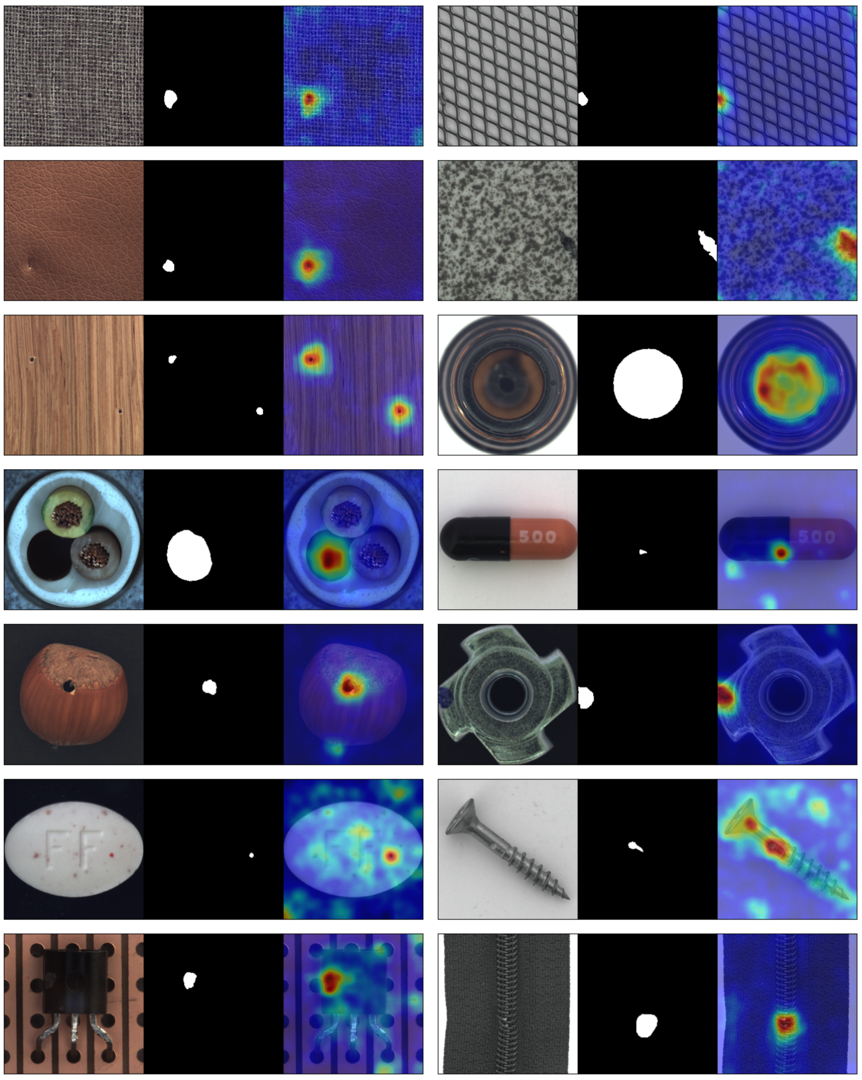


Figure 2.6: Visualization of anomaly segmentation on MVTEC. For each class, from left to right we show one anomalous image, its ground-truth map, and the predicted anomaly heatmap with anomalies highlighted in red. NTL localizes the anomalous regions successfully.

Anomaly segmentation baselines. We compare NTL with five recent anomaly segmentation baselines. They are (i) Patch SVDD [108] that learns DSVDD on the image patches, (ii) CutPaste [88], a self-supervised anomaly segmentation method with hand-crafted local transformations, (iii) Fully Convolutional Data Description (FCDD) [109] that learns one-class spatial features by fine-tuning a pre-trained network, (iv) Semantic Pyramid Anomaly Detection (SPADE) [110] that scores

Table 2.7: Average AUC (%) with standard deviation for anomaly segmentation on MVTEC. NTL achieves the best anomaly segmentation performance on average.

	Class	FCDD	Patch SVDD	CutPaste	SPADE	PaDIM	NTL
Texture	Carpet	96	92.6	98.3±0.0	97.5	**99.1**	98.6±0.01
	Grid	91	96.2	97.5±0.1	93.7	97.3	**98.4±0.05**
	Leather	98	97.4	**99.5±0.0**	97.6	99.2	99.3±0.05
	Tile	91	91.4	90.5±0.2	87.4	94.1	**94.8±0.11**
	Wood	88	90.8	**95.5±0.1**	88.5	94.9	94.1±0.12
	Average	93	93.7	96.3	92.9	96.9	**97.0**
Object	Bottle	97	98.1	97.6±0.1	**98.4**	98.3	97.9±0.03
	Cable	90	96.8	90.0±0.2	**97.2**	96.7	97.2±0.07
	Capsule	93	95.8	97.4±0.1	**99.0**	98.5	98.5±0.01
	Hazelnut	95	97.5	97.3±0.1	**99.1**	98.2	98.3±0.05
	Metal nut	94	98.0	93.1±0.4	98.1	97.2	**98.3±0.14**
	Pill	81	95.1	95.7±0.1	96.5	95.7	**98.3±0.11**
	Screw	86	95.7	96.7±0.1	**98.9**	98.5	98.6±0.04
	Toothbrush	94	98.1	98.1±0.0	97.9	**98.8**	98.4±0.03
	Transistor	88	97.0	93.0±0.2	94.1	**97.5**	94.1±0.09
	Zipper	92	95.1	**99.3**±0.0	96.5	98.5	98.8±0.05
	Average	91	96.7	95.8	97.6	**97.8**	**97.8**
Average		92	95.7	96.0	96.0	97.5	**97.6**

anomalies with Euclidean distance to the normal patch-level features from a pre–trained model, (v) and Patch Distribution Modeling (PaDIM) [111] that fits Gaussian distributions for the patch-level features from a pre-trained model.

2.3.5.2 NTL for Image Anomaly Segmentation

We first process images with a WideResNet50 pre-trained in ImageNet as in previous works [110, 111]. The image features are extracted from the second and third blocks and then concatenated as the patch-level features $\boldsymbol{x} \in \mathbb{R}^{1536}$. The pre-trained feature extractor is frozen. Neural transformations are applied to the patch-level features. $\mathcal{T}_0^\theta$ is modeled by an encoder f^θ. The transformations $\mathcal{T}_k^\theta$ with $k \in \{1, \dots, K\}$ are modeled by composing transformation functions ϕ_k^θ with the shared encoder f^θ. We use the multiplicative parametrization for the transformation function, namely, $\phi_k^\theta(\boldsymbol{x}) := M_k^\theta(\boldsymbol{x}) \odot \boldsymbol{x}$ with $M_k^\theta(\boldsymbol{x}) \in (0,1)^{1536}$. $M_k^\theta(\boldsymbol{x})$ is computed by three bias-free linear layers with affine-free batch normalization layers and ReLU activations. A sigmoid activation is applied to the final layer of M_k^θ. We set the number of neural transformations $K = 15$. The encoder f^θ is modeled by one linear layer with the output dimension of 256. We set the temperature $\tau = 0.1$.

NTL outputs the anomaly score for each patch-level feature. We resize the anomaly heatmap to the image size with the bilinear interpolation and then smooth the resized anomaly heatmap with a Gaussian kernel of the width 4.

2.3.5.3 Empirical Results

We report the results in terms of pixel-level AUC (%) in Table 2.7. The results of baselines are taken from their original publications [88, 108, 109, 110, 111]. NTL achieves the best anomaly segmentation performance on average. Recent work builds shallow anomaly detection mechanisms, such as KNN and KDE, upon patch-level features from pre-trained models [110, 111]. However, generalizing self-supervised approaches to image features is not straightforward. To this end, NTL applies learnable transformations on patch-level features, yielding better results than existing baselines using pre-trained models [109, 110, 111]. Compared to CutPaste [88], the self-supervised baseline, NTL learns the transformations from data rather than designs them manually. This allows for the application of NTL directly on image features. Consequently, NTL outperforms CutPaste with hand-crafted transformations and saves the cost of designing the transformations and training the model from scratch.

We also visualize the predicted anomaly heatmaps in Figure 2.6. NTL outputs anomaly heatmaps (right) that match the ground-truth maps (middle) very well. Even though some anomalous regions, such as in the images of pill and capsule, are even hard for humans, NTL localizes them successfully.

2.3.6 Anomaly Detection on Text

There are many beneficial applications of anomaly detection on text, such as detecting spam emails, fake tweets, or other anomalous content on the web. Here we present NTL for text anomaly detection and compare it with recent approaches for sentence-level anomaly detection. NTL on text uses a pre-trained language model to preprocess the text. To create different views of each sentence, neural transformations are then applied to the list of word embeddings comprising each sentence. Below, we describe how the individual word embeddings are transformed and how an attention mechanism is used to aggregate word embeddings to create the different views of each sentence needed for the NTL loss.

2.3.6.1 Datasets and Baselines

Text datasets. We study NTL on the *Reuters-21578* dataset, which is commonly used in previous text anomaly detection works [13, 23, 26]. We use the same preprocessing as in Ruff et al. [23], including lowercasing, removing stopwords, and tokenization. As in Ruff et al. [23], we select seven classes that have exactly one label. They are *earn*, *acq*, *crude*, *trade*, *money-fx*, *interest*, and *ship*. We evaluate NTL on them with the one-vs-rest setting.

Text baselines. The text anomaly detection baselines include OCSVM [13] and DSVDD [89], which aim to encompass normal data within a hypersphere in the embedding space, and Multi-modal Deep Support Vector Data Description (mSVDD) [26] which extends the idea of DSVDD to multiple hyperspheres. Further, the baselines include Deep Multi-sphere Support Vector Data Description (DMSVDD) [25], a hard version of mSVDD, and Context Vector Data Description (CVDD) [23], which augments DSVDD for text with multiple attention heads to reflect that there are multiple normal semantic contexts for each word. All baselines are built upon the same pre-trained language model GloVe 6B [112].

2.3.6.2 NTL on Text

NTL on text uses a pre-trained language model to preprocess the text. To create different views of each sentence, neural transformations are then applied to the list of word embeddings comprising each sentence. Below, we describe how the individual word embeddings are transformed and how an attention mechanism is used afterward to aggregate the word embeddings to create the different views of each sentence.

As in previous work [23, 25, 26], we preprocess the data with a language model such that each sentence $\boldsymbol{x}$ is represented by a sequence of word embeddings $\boldsymbol{x} = [\boldsymbol{e}_1, \ldots, \boldsymbol{e}_T] \in \mathbb{R}^{d\times T}$. The length T of the sentences varies from sentence to sentence. $\mathcal{T}_0^\theta$ is modeled by an encoder f^θ. The transformations $\mathcal{T}_k^\theta$ with $k \in \{1, \cdots, K\}$ are modeled by composing transformation functions ϕ_k^θ with the shared encoder f^θ. The transformation function ϕ_k^θ is applied to the individual word embeddings, producing the transformed embeddings,

$$\boldsymbol{x}_k = \phi_k^\theta(\boldsymbol{x}) = M_k^\theta(\boldsymbol{x}) \odot \boldsymbol{x},$$

where M_k^θ is a feed-forward neural network with a sigmoid activation on the top. It is applied to each embedding in the sentence separately, i.e., $M_k^\theta(\boldsymbol{x}) = [M_k^\theta(\boldsymbol{e}_1), \cdots, M_k^\theta(\boldsymbol{e}_T)] \in (0,1)^{d\times T}$. After the element-wise multiplication with word embeddings, we have the transformed embedding $\boldsymbol{x}_k = [M_k^\theta(\boldsymbol{e}_1) \odot \boldsymbol{e}_1, \cdots, M_k^\theta(\boldsymbol{e}_T) \odot \boldsymbol{e}_T] \in \mathbb{R}^{d\times T}$. The embeddings are aggregated using attention $A^\theta(\boldsymbol{x}) \in (0,1)^T$ and are then encoded. In summary,

$$f^\theta(\boldsymbol{x}_k) = g^\theta(\boldsymbol{x}_k A^\theta(\boldsymbol{x})) \quad \text{with} \quad A^\theta(\boldsymbol{x}) = \mathrm{sigmoid}(\tanh(\boldsymbol{x}^\top \boldsymbol{w}_1)\boldsymbol{w}_2), \tag{2.18}$$

where $\boldsymbol{w}_1$ and $\boldsymbol{w}_1$ are learnable weights, and g^θ is a feed-forward neural network that maps the aggregated embeddings to a low dimensional embedding space [5].

[5]for the untransformed sentence $\boldsymbol{x}$, Equation (2.18) becomes $f^\theta(\boldsymbol{x}) = g^\theta(\boldsymbol{x}A^\theta(\boldsymbol{x}))$.

Table 2.8: Average AUC (%) with standard deviation on Reuters datasets. NTL achieves the best average AUC and the best accuracy on 5 of 7 experimental variants.

Class	OCSVM	DSVDD	CVDD	DMSVDD	mSVDD	NTL
earn	91.1	95.9	91.8	96.0	95.9	**97.4±0.1**
acq	**93.1**	89.4	91.5	89.8	90.1	91.6±1.1
crude	92.4	92.8	95.5	92.1	92.4	**96.4±0.4**
trade	99.0	98.4	99.2	98.8	98.6	**99.5±0.1**
money-fx	88.6	86.3	82.8	87.1	87.1	**89.8±0.1**
interest	97.4	97.3	97.7	97.2	97.3	**98.6±0.4**
ship	91.2	92.5	**95.6**	93.0	92.6	89.6±0.7
average	93.3	93.2	93.1	93.4	93.4	**94.7±0.1**

Table 2.9: Example of relevant words (with large attention weights) for each class. The attention mechanism learns to highlight words with a similar semantic content.

classes	earn	acq	crude	trade	money-fx	interest	ship
	pretax	*share*	*futures*	*imports*	*dollar*	*rate*	*ship*
x	*rupees*	*dollars*	*crude*	*exports*	*currency*	*pct*	*ships*
	dlrs	*billion*	*oil*	*trade*	*bank*	*rates*	*port*

Implementation details. For M_k^θ we use a neural network of three bias-free linear layers of units $[300, 300, 300]$ with ReLU activations and affine-free batch normalization layers and a sigmoid activation on the top. The weights $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ have the shapes of $[300, 300]$ and $[300, 1]$. For g^θ we use a neural network of three bias-free linear layers of units $[300, 100, 100]$ with ReLU activations. We set the number of transformations $K = 10$ and the temperature $\tau = 0.1$.

2.3.6.3 Empirical Results

The detection results in terms of average AUC (%) with standard deviation are reported in Table 2.8. While all existing baselines apply one-class classification-based approaches to text anomaly detection, NTL explores the application of self-supervised detection on the text and achieves the best detection accuracy on 5 of 7 experimental variants. On average, NTL outperforms all baselines by at least 1.3%.

We also record the three most relevant words (in terms of highest attention weight $A^\theta(\boldsymbol{x})$) for each class in Table 2.9. We can see that the attention mechanism of NTL learns to highlight words with similar semantic content. This can give hints on selecting the signature words in specific applications.

2.4 Summary and Discussion

We propose NTL, a self-supervised anomaly detection method with learnable transformations. The key ingredient is a novel training objective, DCL, which encourages the learned transformations to be diverse and semantically meaningful. This unleashes the power of self-supervised anomaly detection to general data types. We theoretically demonstrate that DCL is more suited than existing self-supervised loss functions for transformation learning and anomaly detection. We also critically discuss the theoretical limitations of NTL on learning transformations and detecting anomalies. Addressing these, we give practical recommendations on the architecture design and implementation. Our extensive empirical study finds that on various data types, including time series, images, tabular data, and text data, learning transformations and detecting anomalies with NTL improves over existing anomaly detection approaches. Beyond anomaly detection, we also show that NTL can better localize the anomalous regions of an image than many existing methods.

The idea of learning data augmentation schemes is also studied in other domains previously. "AutoAugmentation" has usually relied on composing hand-crafted data augmentations [113, 114, 115, 116, 117]. Tran et al. [118] learn Bayesian augmentation schemes for neural networks, and Wong and Kolter [119] learn perturbation sets for adversarial robustness. Though their setting and approach are different, our work is most closely related to Tamkin et al. [120], which studies how to generate views for representation learning in the framework of Chen et al. [73]. They parameterize their "viewmakers" as residual perturbations, which are trained adversarially to avoid trivial solutions where the views share no semantic information with the original sample. In contrast, under NTL, there are no restrictions on the architecture of the transformations, as long as the DCL can be optimized (i.e., the gradient is well defined). The DCL is an adequate objective for training the transformations jointly as it manages the trade-off between semantics and diversity.

Parts of this chapter are mainly based on:

[121] **Chen Qiu**, Timo Pfrommer, Marius Kloft, Stephan Mandt, and Maja Rudolph. Neural Transformation Learning for Anomaly Detection beyond Images. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of Proceedings of Machine Learning Research, pages 8703–8714. PMLR, 18–24 Jul 2021.

Chen Qiu, Marius Kloft, Stephan Mandt, and Maja Rudolph. Self-Supervised Anomaly Detection with Neural Transformations. *Preprint (under review)*, 2022.

3 Extensions of Neural Transformation Learning

In this chapter, we introduce the extensions of NTL and show that NTL is compatible with representation learning and One-Class Classification (OCC) learning. Benefiting from combining NTL with other learning paradigms, our methods achieve impressive results in anomaly detection within time series and graph-level anomaly detection.

In Section 3.1 we study detecting anomalies within time series, which is essential in many application domains, reaching from self-driving cars, finance, and marketing to medical diagnosis and epidemiology. Addressing this, we develop Local Neural Transformations (LNT), a method for learning local transformations of time series and producing an anomaly score for each time step to detect anomalies within the time series. LNT combines NTL and time series representation learning, Contrastive Predictive Coding (CPC), which learns representations that capture both local and contextualized semantics. However, the representation learning method is not suitable for detecting anomalies. To do so, NTL is built upon the representations of each time step for anomaly detection. In theory, we prove that both components complement each other to avoid trivial solutions not appropriate for anomaly detection within time series. Our experimental evaluations show that LNT can detect anomalies in time series with complex dynamics.

In Section 3.2, we study graph-level anomaly detection, which has become a critical topic in diverse areas, such as financial fraud detection and detecting anomalous activities in social networks. In graph-level anomaly detection, we are interested in detecting entire abnormal graphs in a set of graphs. Addressing this, we introduce One-Class Graph Transformation Learning (OCGTL) and Multi-view One-Class Classification (MOCC). We show that both OCGTL and MOCC complement deep OCC with learnable transformations to be more powerful without exhibiting hypersphere collapse. On the other hand, deep OCC also complements NTL to exploit better the graph size information in graph-level anomaly detection. Our extensive experiments demonstrate that OCGTL and MOCC raise the bar in graph-level anomaly detection significantly.

3.1 Local Neural Transformation Learning

Anomaly detection within time series is significant in many industrial, medical, and scientific applications. For instance, undetected anomalies in water treatment facilities or chemical plants can bring harm to millions of people. Such systems need to be constantly monitored for anomalies. Existing methods for detecting anomalies within time series using deep learning include forecasting methods [122, 123, 124, 125], autoencoder-based methods [105, 126, 127, 128], and deep generative models [129, 130, 131, 132, 133, 134, 135, 136]. Although there has been growing interest in tackling anomaly detection with self-supervised learning, there is limited success of self-supervised methods in detecting anomalies within time series.

NTL learns transformations and makes self-supervised anomaly detection applicable to general data types. While this approach can identify an entire sequence as anomalous (Section 2.3), it can still not be applied to detecting anomalies *within* time series. NTL can lead to trivial transformations that are not suitable for anomaly detection within time series. For anomaly detection within time series, both local semantics (the dynamics within a time window) and contextualized semantics (how the time window relates to the remaining time series) matter. To this end, we propose LNT, a new framework for detecting anomalies within time series data that combines NTL and time series representation learning.

We first present the algorithm in Section 3.1.1. Then, we provide theoretical arguments for combining transformation learning and representation learning in Section 3.1.2 and show that both learning paradigms complement each other to avoid trivial solutions not appropriate for detecting anomalies. In the end, our experiments in Section 3.1.3 demonstrate that LNT can find anomalies in speech segments from the LibriSpeech dataset and detect interruptions to cyber-physical systems.

3.1.1 Algorithm and Methodology of LNT

LNT has two components: a feature extractor (encoder) and an anomaly detector (local neural transformations). Given an input sequence, an encoder produces an embedding for each time step, encoding relevant information from the current time window. These features are then transformed by applying distinct neural networks to each embedding, producing different *latent views*. The views are trained to fulfill two requirements; the views should be diverse and semantically meaningful, i.e., they should reflect both local dynamics as well as how the observations fit into the larger context of the time series. The requirements are encouraged via self-supervision.

Specifically, two aspects of LNT are self-supervised – it combines two different contrastive losses. One of the contrastive losses, CPC, guides the representation learning that guarantees the encoder of LNT to produce meaningful semantic time

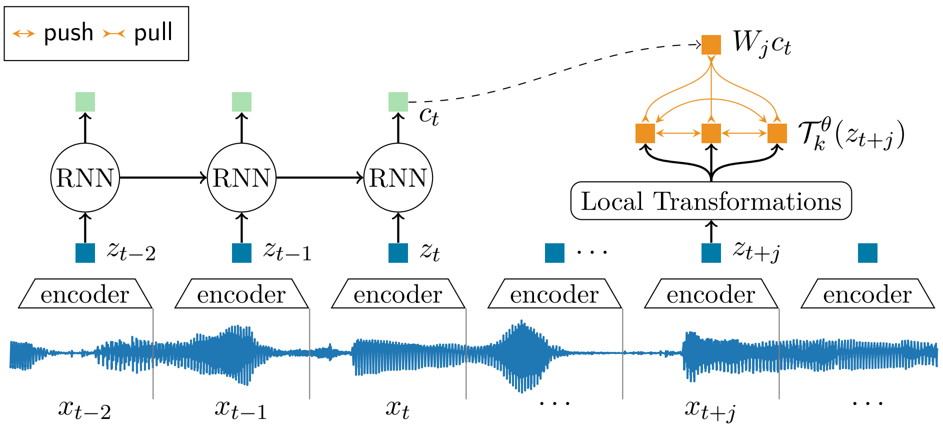


Figure 3.1: Overview of LNT: neural transformations are built upon the local representations $\boldsymbol{z}_t$ at each time step to generate a variety of latent views $\mathcal{T}_k^\theta(\boldsymbol{z}_t)$. Neural transformations and the encoder (and RNN) are jointly trained with a unified loss combining the CPC loss and the DDCL.

series representations that generalize well to unseen test data. The second contrastive loss, a novel Dynamic Deterministic Contrastive Loss (DDCL), guides the transformation learning to learn a variety of latent views being semantically representative of the time series, both in a local and in a contextualized sense.

LNT follows the general paradigm of self-supervised anomaly detection. During training, the capability to generate diverse and semantically meaningful data views improves for normal data, while it does not generalize to anomalies. Given a (potentially multivariate) time series $\boldsymbol{x}_{1:T}$ with $\boldsymbol{x}_t \in \mathbb{R}^d$, our method should output scores for each individual time step, representing the likelihood that the observation in this time step is an anomaly. The inputs are processed by an encoder that is trained jointly with the local neural transformations. Before presenting local transformation learning, we describe the learning of local time series representations.

3.1.1.1 Local Time Series Representations

The first part of the LNT architecture is an encoder (as shown in Figure 3.1). The encoder maps a sequence of samples to a sequence of local latent representations $\boldsymbol{z}_t$ and is trained using the principles of CPC [62]. We use the same architecture as Oord et al. [62]. The representations produced by the encoder $\boldsymbol{z}_t = g^\theta_{\text{enc}}(\boldsymbol{x}_t)$ are summarized with an autoregressive module into context vectors $\boldsymbol{c}_t = g^\theta_{\text{ar}}(\boldsymbol{z}_{\leq t})$. We have a function measuring the alignment of the representation of $\boldsymbol{x}_t$ and the linear

j-step prediction $\boldsymbol{w}_j \boldsymbol{c}_t$[1] as

$$h_j(\boldsymbol{x}_t, \boldsymbol{c}_t) = \exp(\boldsymbol{z}_t^\top \boldsymbol{w}_j \boldsymbol{c}_t) \tag{3.1}$$

For all t and all prediction steps j, we sample a set $\mathcal{M}$ from the training set $\mathcal{D}$ that contains one positive pair $(\boldsymbol{x}_t, \boldsymbol{x}_{t+j})$ and negative samples in the rest. The CPC loss contrasts $\boldsymbol{x}_{t+j}$ to negative samples given the linear j-step future prediction $\boldsymbol{w}_j \boldsymbol{c}_t$ as the anchor:

$$\mathcal{L}_{\mathrm{CPC}}^{\theta} = -\mathbb{E}_{\mathcal{M}\sim\mathcal{D}}\left[\log \frac{h_j(\boldsymbol{x}_{t+j}, \boldsymbol{c}_t)}{\sum_{\boldsymbol{x}_i \in \mathcal{M}} h_j(\boldsymbol{x}_i, \boldsymbol{c}_t)}\right]. \tag{3.2}$$

It encourages the context representation $\boldsymbol{c}_t$ to be predictive of nearby local representations of $\boldsymbol{x}_{t+j}$. Optimizing Equation (3.2) relates to maximizing the mutual information [66] between the context representation $\boldsymbol{c}_t$ and nearby time points $\boldsymbol{x}_{t+j}$ to produce good representations ($\boldsymbol{z}_t$ and $\boldsymbol{c}_t$) that can be used in the downstream tasks, including anomaly detection.

3.1.1.2 Local Neural Transformations

The second part of the LNT architecture (above the encoder as shown in Figure 3.1) introduces an auxiliary task for learning and anomaly detection. The time series representations $\boldsymbol{z}_t$ are processed by local neural transformations to produce different views of each embedding. This operation relates to data augmentation but has two major differences: First, the transformations are not applied at the data level but in the latent space, producing *latent views* of each time window. Second, the transformations are not hand-crafted as is often done in computer vision but are directly learned during training.

The neural transformations are K neural networks $\mathcal{T}_k^\theta$ with parameters θ for $k = 1, \dots, K$. They are applied to each latent representation $\boldsymbol{z}_t$ to produce different latent views $\mathcal{T}_k^\theta(\boldsymbol{z}_t)$. Each of the transformed views is encouraged to be predictive of the context at different time horizons j by a loss contribution

$$\ell_t^{(j,k)}(\boldsymbol{x}_{\leq t}; \theta) = -\log \frac{h(\mathcal{T}_k^\theta(\boldsymbol{z}_t), \boldsymbol{w}_j \boldsymbol{c}_{t-j})}{h(\mathcal{T}_k^\theta(\boldsymbol{z}_t), \boldsymbol{w}_j \boldsymbol{c}_{t-j}) + \sum_{l \in \{1,\dots,K\}/\{k\}} h(\mathcal{T}_k^\theta(\boldsymbol{z}_t), \mathcal{T}_l^\theta(\boldsymbol{z}_t))},$$

which simultaneously pushes different views of the same latent representations apart from each other. The loss contributions of each time-step t, each transformation k, and each time horizon j are combined to produce the Dynamic Deterministic Contrastive Loss (DDCL):

$$\mathcal{L}_{\mathrm{DDCL}}^{\theta} = \mathbb{E}_{\boldsymbol{x}_{1:T}\sim\mathcal{D}}\left[\sum_{j=1}^{J}\sum_{t=1}^{T}\sum_{k=1}^{K} \ell_t^{(j,k)}(\boldsymbol{x}_{\leq t}; \theta)\right]. \tag{3.3}$$

[1] $\boldsymbol{w}_j$ is a learnable parameter included in θ.

During training, the two objectives (Equations (3.2) and (3.3)) are optimized jointly using a unified loss,

$$\mathcal{L}^{\theta}_{\mathrm{LNT}} = \mathcal{L}^{\theta}_{\mathrm{CPC}} + \lambda \cdot \mathcal{L}^{\theta}_{\mathrm{DDCL}} \tag{3.4}$$

and a balancing hyperparameter λ. We set $\lambda = 1$ in default.

As depicted by orange arrows in Figure 3.1, $\mathcal{L}^{\theta}_{\mathrm{DDCL}}$ can intuitively be interpreted as pushing and pulling different representations in latent space. The numerator pulls the learned transformations $\mathcal{T}^{\theta}_{k}(\boldsymbol{z}_{t+j})$ close to $\boldsymbol{w}_j \boldsymbol{c}_t$ ensuring semantic views, while the denominator pushes different views apart, ensuring diversity in the learned transformations.

After training LNT on a dataset of typical time series, we can use the DDCL for anomaly detection. Given a test sequence $\boldsymbol{x}_{1:T}$, we evaluate the contribution of individual time steps to $\mathcal{L}^{\theta}_{\mathrm{DDCL}}$ (Equation (3.3)). The score for each time point t in the sequence is,

$$s_t(\boldsymbol{x}_{\leq t}) = \sum_{j=1}^{J} \sum_{k=1}^{K} \ell^{(j,k)}_{t}(\boldsymbol{x}_{\leq t}; \theta) \tag{3.5}$$

The higher the score, the more likely the series exhibits abnormal behavior at time t. Unlike CPC-based anomaly detection [137], this anomaly score has the advantage of being deterministic, and thus there is no need to draw negative samples from a proposal or noise distribution.

3.1.2 Theoretical Properties of LNT

Since the LNT architecture (shown in Figure 3.1) is trained on two losses jointly (the CPC loss and the DDCL), the natural question arises: are both losses necessary, or could we just train on the DDCL alone? The following analysis demonstrates the value of considering both losses jointly.

The following theorem shows that, if we trained the LNT architecture (i.e., the encoder and transformations $\mathcal{T}^{\theta}_{k}$) only on the $\mathcal{L}^{\theta}_{\mathrm{DDCL}}$ (without the $\mathcal{L}^{\theta}_{\mathrm{CPC}}$), the optimal solution would collapse to a constant encoder. Thus $\mathcal{L}^{\theta}_{\mathrm{CPC}}$ acts as a regularizer in our LNT learning framework to avoid the learning of an uninformative constant encoder; it is thus strictly necessary.

Proposition 7. Let g^{θ}_{enc} and g^{θ}_{ar} be arbitrary encoders (including biases) with learned parameters θ, and let $\mathcal{L}^{\theta}_{\mathrm{DDCL}}$ be the corresponding DDCL. Then there exist constant encoders $g^{\tilde{\theta}}_{enc}$ and $g^{\tilde{\theta}}_{ar}$ (i.e., $\exists \tilde{\theta}, \boldsymbol{a}, \boldsymbol{b}\ \forall \boldsymbol{x}, \boldsymbol{z} : g^{\tilde{\theta}}_{enc}(\boldsymbol{x}) = \boldsymbol{a}, g^{\tilde{\theta}}_{ar}(\boldsymbol{z}) = \boldsymbol{b}$) with

$$\mathcal{L}^{\tilde{\theta}}_{\mathrm{DDCL}} \leq \mathcal{L}^{\theta}_{\mathrm{DDCL}}.$$

Proof. Let g^{θ}_{enc} and g^{θ}_{ar} be arbitrary encoders (including biases) with learned parameters θ, and let $\mathcal{L}^{\theta}_{\mathrm{DDCL}}$ be the corresponding DDCL. We observe from Equation (3.3) that $\mathcal{L}^{\theta}_{\text{DDCL}}$ decomposes into a sum of loss contributions $\ell_t^{(j,k)}(\boldsymbol{x}_{\leq t};\theta)$. Let

$$(\boldsymbol{x}^*_{\leq t^*}, j^*, t^*) = \arg\min \sum_{k=1}^{K} \ell_t^{(j,k)}(\boldsymbol{x}_{\leq t};\theta), \tag{3.6}$$

be the indices of the summands with the smallest contribution to the sum, for a given fixed θ. This means $\boldsymbol{x}^*$ is the sample, j^* the time horizon, and t^* the time point associated with the smallest loss contribution to $\mathcal{L}_{\text{DDCL}}$. Put

$$\ell^* := \sum_{k=1}^{K} \ell_t^{(j^*,k)}(\boldsymbol{x}_{\leq t^*};\theta). \tag{3.7}$$

Since our encoders are equipped with bias terms there exist constant encoders $g^{\tilde{\theta}}_{enc}$ and $g^{\tilde{\theta}}_{ar}$ (i.e., $\exists \tilde{\theta}, \boldsymbol{a}, \boldsymbol{b} \,\forall \boldsymbol{x}, \boldsymbol{z} : g^{\tilde{\theta}}_{enc}(\boldsymbol{x}) = \boldsymbol{a}, g^{\tilde{\theta}}_{ar}(\boldsymbol{z}) = \boldsymbol{b}$) with

$$\forall \boldsymbol{x}, j, t : \sum_{k=1}^{K} \ell_t^{(j,k)}(\boldsymbol{x}_{\leq t};\tilde{\theta}) = \ell^*. \tag{3.8}$$

Then we have:

$$\begin{aligned}
\mathcal{L}^{\theta}_{\text{DDCL}} &\overset{(3.3)}{=} \mathbb{E}\left[\sum_{j=1}^{J}\sum_{t=1}^{T}\sum_{k=1}^{K} \ell_t^{(j,k)}(\boldsymbol{x}_{\leq t};\theta)\right] \\
&\overset{(3.6)}{\geq} JT \sum_{k=1}^{K} \ell_t^{(j^*,k)}(\boldsymbol{x}^*_{\leq t^*};\theta) \overset{(3.7)}{=} JT\ell^* \\
&\overset{(3.8)}{=} \mathbb{E}\left[\sum_{j=1}^{J}\sum_{t=1}^{T}\sum_{k=1}^{K} \ell_t^{(j,k)}(\boldsymbol{x}_{\leq t};\tilde{\theta})\right] \overset{(3.3)}{=} \mathcal{L}^{\tilde{\theta}}_{\text{DDCL}},
\end{aligned}$$

which was to prove. □

The above theorem shows that if LNT was trained on the DDCL only, LNT would collapse into a trivial solution. On the other hand, a constant encoder clearly does not optimize the maximum mutual information criterion [62], which is induced by the CPC objective.

Besides this hard mathematical evidence, there are also other good reasons to include the CPC loss in LNT. For instance, it ensures that the latent representations account for dynamics at longer time scales. This task is carried out by CPC's autoregressive module. Our hypothesis is that, for effective anomaly detection within time series, it is necessary to consider both: the local signal in a time window and the larger context across time windows. Otherwise, the observations within a time window could be perfectly normal while not making sense in the context of a longer time horizon. For this reason, we believe that there are two types of semantic requirements of the representations and the latent views of LNT:

- *Local semantics*: views should share relevant semantic information with the current time window. (This is addressed by $\mathcal{L}^{\theta}_{\mathrm{CPC}}$)
- *Contextualized semantics*: views should reflect how the time window relates to the rest of the time series at different, longer time horizons. (This is addressed by $\mathcal{L}^{\theta}_{\mathrm{DDCL}}$)

Both loss contributions of LNT facilitate these requirements. CPC contributes local latent representations and context representations. The semantic content of the views is managed by the DDCL. Especially its numerator ensures contextualized semantics: the view $\mathcal{T}^{\theta}_{k}(\boldsymbol{z}_t)$ should be close to the context information $\boldsymbol{w}_j\boldsymbol{c}_{t-j}$ with various lags j. This gives the LNT architecture a lever to consider longer time horizons from different (non-transformed) contexts $\boldsymbol{c}_{t-j}$ when deciding whether there is an anomaly at time t meaning that it also exhibits contextual semantic.

3.1.3 Experiments on Anomaly Detection within Time Series

In the experimental evaluation, we compare LNT to existing strong baselines on challenging time series anomaly detection datasets. We first describe the datasets, baselines, and implementation details. Second, we present our findings: LNT outperforms many strong baselines in detecting anomalies in the operation of a water treatment system and accurately finds anomalies in speech. Third, we provide visualizations of the local transformations that are learned by LNT. Finally, we analyze the performance of LNT in comparison to CPC-based alternatives.

3.1.3.1 Datasets

We evaluate LNT on two challenging real-world datasets, namely, the Secure Water Treatment dataset (SWaT) [138] and the Libri Speech Collection [139]. The SWaT dataset is provided with labeled anomalies in the test set. The LibriSpeech data is augmented with realistic synthetic anomalies.

SWaT. This dataset is from a testbed for water treatment [140] that evaluates the cyber-security of a fully functional plant with a six-stage process of filtration and chemical dosing. Goh et al. [138] collected eleven days of operation data. Under normal operation, 51 sensor channels are recorded for seven days yielding a training time series of the length of 475200. For the test data of the length of 224960, 36 attacks were launched during the last four days of the collection process. As suggested in Goh et al. [138], Li et al. [141], the first 21600 samples from the training data are removed for training stability. We follow the experimental setup of He and Zhao [124] and take the first part of the collection under attack as the validation set. We also drop channels that are constant in both training and test set, yielding a time series of 45 dimensions.

LibriSpeech. The LibriSpeech dataset [139] is an audio collection with spoken language recordings from 251 distinct speakers. We adopt the setup of Oord et al. [62] and their train/test split. For benchmarking anomaly detection performance, we randomly place additive pure sine tones of varying frequency (20 - 120 Hz) and length (512 - 4096 time steps) in the test data, yielding consecutive anomaly regions making up $\approx 10\%$ of the test data. Speech data offers a challenging benchmark for anomaly detection within time series since speech data typically exhibits complex temporal dynamics due to its high multi-modality introduced through different speakers and word sequences.

3.1.3.2 Baselines

We study LNT in comparison to different classes of anomaly detection algorithms, ranging from classical methods to recent advances in deep anomaly detection. They include (i) methods that leverage the reconstruction of the autoencoder or variational autoencoder, including LSTM-ED [105], LSTM-VAE [130], and Multi-Scale Convolutional Recurrent Encoder-Decoder (MSCRED) [127]. (ii) methods that use the ability of GANs to discriminate fake examples, like BeatGAN [142] and MadGAN [141], (iii) OmniA [133] which estimates the data density with a stochastic recurrent neural network, (iv) and Temporal Hierarchical One-Class Network (THOC) [143] for time-series which employs a one-class objective.

Implementation details. For LNT, the hyperparamaters are adopted from those reported by Oord et al. [62] for CPC: especially $\boldsymbol{c}_t \in \mathbb{R}^{256}$ and $\boldsymbol{z}_t \in \mathbb{R}^{512}$ for experiments with LibriSpeech data. The data is processed in sub-sequences of length 20480 for both training and testing. Since the SWaT dataset contains way less diverse data points and shows simpler temporal dynamics, the embeddings sizes are reduced to $\boldsymbol{c}_t \in \mathbb{R}^{32}$ and $\boldsymbol{z}_t \in \mathbb{R}^{128}$. Also, the convolutional encoder network is downsized to filters $(3, 3, 4, 2)$ and strides $(3, 3, 4, 2)$ resulting in the convolution of 72 time steps. We consistently choose $K = 12$ learnable transformations on both datasets. Each transformation is parametrized multiplicatively as $\mathcal{T}_k^\theta(\boldsymbol{z}_t) = \boldsymbol{z}_t \odot M_k^\theta(\boldsymbol{z}_t)$, where $M_k^\theta(\boldsymbol{z}_t)$ is an attention mask (with values between 0 and 1) and the multiplication is applied element-wise. M_k^θ is modeled by a two-layer MLP on the SWaT and a three-layer MLP on the LibriSpeech data. All MLPs have intermediate ReLU activations and a sigmoid activation on the top. The final layer always shares the dimensionality of $\boldsymbol{z}_t$. All bias terms in the network are fixed as zero.

Table 3.1: F1-score (%) for anomaly detection on the SWaT. Baseline results are taken from Shen et al. [143]. LNT achieves the best F1-score and the best recall.

	LSTM-ED	LSTM-VAE	MadGAN	BeatGAN	OmniA	MSCRED	THOC	LNT
Prec	93.69	98.39	98.72	88.37	**99.01**	98.43	98.08	94.75
Rec	63.31	77.01	77.60	76.41	77.06	77.69	79.94	**83.28**
$F1$	75.56	86.39	86.89	81.95	86.67	86.84	88.09	**88.65**

3.1.3.3 Quantitative Results

We judge the anomaly scores predicted by the algorithms for each time step individually. The results on the SWaT datasets in terms of F1-score are reported in Table 3.1. LNT outperforms a set of challenging baselines reported in Shen et al. [143] with the highest F1-score (88.65%) and the highest recall (83.28%). A high recall reflects a better performance in uncovering some of the harder detectable anomalies. In many mission-critical applications, detecting as many anomalies as possible with reasonable precision is often critical, as a false negative can do more harm than a false positive. This makes the high recall of LNT preferable.

We also compare LNT with recent deep learning methods on the LibriSpeech data, which has more complex temporal dynamics. Successful detection of these complex anomalies requires a deep understanding of their temporal dynamics. We report the results in terms of ROC curves in Figure 3.2. Here, LNT clearly outperforms both deep learning methods. This shows that detecting anomalies within speech data with its complex temporal dynamics is indeed a challenging task for many deep anomaly detection algorithms. Especially the future predictions of LSTM-ED perform only slightly better than random chance in this experiment for all possible thresholds. This emphasizes the benefit of building the anomaly detection method upon self-supervised representation learning. We also provide the visualizations of LNT detection on LibriSpeech in Appendix A.3.

3.1.3.4 Visualization of Transformations

In general, it is considered hard to get insights from embedding visualizations for $\boldsymbol{z}_t$ in the latent space. Hence, to make the transformations interpretable in terms of semantics, we propose to visualize them in data space. We reuse the encoder as described in Section 3.1.1 and enrich it with a separate decoder. We train the decoder to reconstruct the (non-transformed) input data while freezing the encoder weights. The trained decoder is then applied to transformed embeddings to visualize them in data space.

We visualize four transformations showing interpretable behaviors on the SWaT dataset in Figure 3.3. For the non-transformed series $\boldsymbol{x}$ the signal jumps in channels 25 and 36 at $t \approx 2500$. This jump is delayed for channels $26 - 35$. Interestingly,

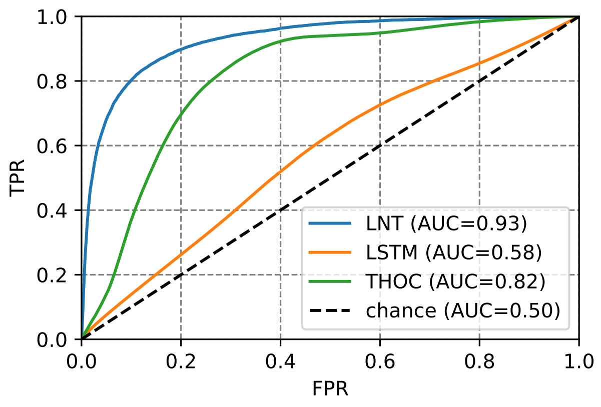


Figure 3.2: Our approach LNT outperforms deep time series anomaly detection baselines on LibriSpeech data in terms of ROC curves.

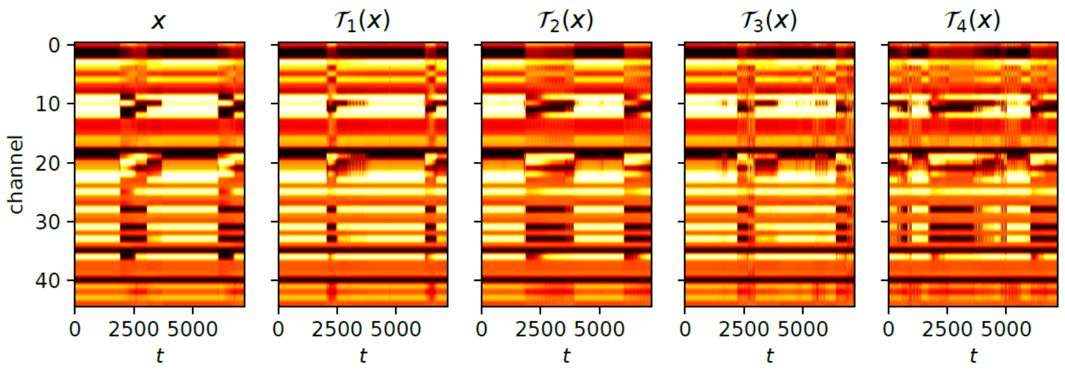
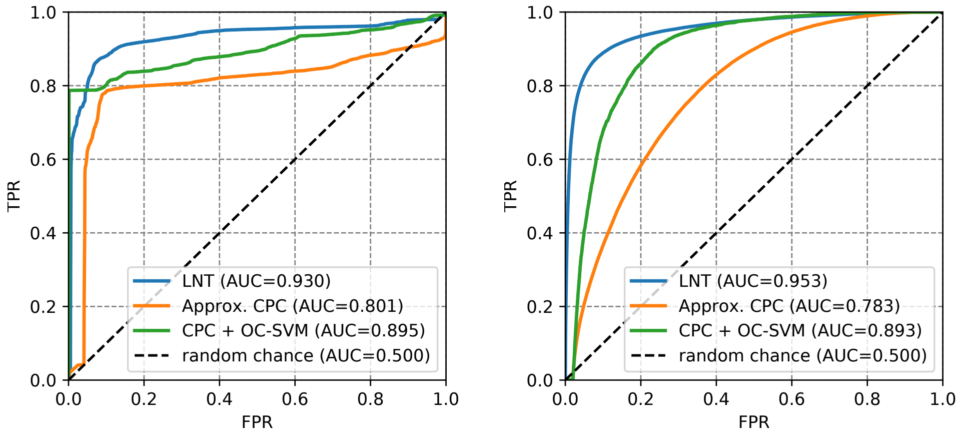


Figure 3.3: Data space visualizations of learned transformations on the SWaT dataset. For visualization, the representations are mapped to the data space with a separately trained decoder. The transformations show semantically interpretable behavior, such as altered delays in specific channels.

we observe that this delay is altered by the learned transformations. For example, $\mathcal{T}_1$ removes this delay causing the signal jump for all aforementioned channels at $t \approx 2500$. In contrast, $\mathcal{T}_2$ affects the series oppositely by enlarging this delay.

In summary, these transformations produce semantically meaningful and diverse views of the time series. Admittedly, current interpretations are still rather high-level and fairly limited from application standpoints. However, without domain knowledge, there exists no gold standard for a good time series transformation to compare against. This was the original motivation for the usage of learnable transformations as effective data augmentation for anomaly detection.



(a) SWaT (b) LibriSpeech

Figure 3.4: Improvement of LNT over CPC-based scoring evaluated on SWaT and LibriSpeech datasets. Our proposed method LNT (combing transformation learning and CPC) consistently outperforms the other variants of CPC for anomaly scoring.

3.1.3.5 Comparison to CPC-based Anomaly Detection

Finally, we study the advantage of LNT over CPC on anomaly detection within time series. There are various ways to use CPC for anomaly detection. Beyond LNT, we consider two methods that build on CPC: (i) methods that directly use the CPC loss to score anomalies [137] and (ii) methods that use CPC as a feature extractor and then run another anomaly detection method such as OCSVM on the extracted features. One disadvantage of (i) is the sampling of negative samples. This makes it nontrivial to evaluate the CPC loss on test data. de Haan and Löwe [137] argue that taking samples from the test data is biased and using the training data is infeasible in practice. Alternatively, we use the numerator of the CPC loss (without counting the negative samples) as the anomaly score at test time.

In contrast, the DDCL is deterministic, and the views are constructed from a single sample. It is hence straightforward to use it to score anomalies. From the results in Figure 3.4, we can see that the combination of transformation learning with representation learning of CPC consistently outperforms the considered variants of CPC for anomaly detection on both datasets. This connects to the discussion about contextualized semantics in Section 3.1.2. Comparing LNT with CPC + OCSVM supports our claim: while OCSVM with CPC input features has access only to the local semantics in the local representations, LNT learns transformations exhibiting both local and contextualized semantics, leading to consistently superior anomaly detection performance.

3.2 One-Class Graph Transformation Learning

Many web-based systems are best represented by graphs and there have been many works on detecting anomalous nodes and edges within a graph [144]. However, in many applications, it is much more relevant to ask whether an entire graph is abnormal. For example, one might be interested in detecting novel (anomalous) molecules whose atoms and bonds are nodes and edges in the molecular graphs. Thus, while every atom may be known, the molecule might be novel as a whole. Furthermore, graph-level anomaly detection could be very useful in financial fraud detection. Consider a graph where nodes represent entities and edges represent financial transactions. A money-laundering scheme involving many parties would not be detectable on a node level but would require identifying the whole graph (or at least a segment of it) as anomalous. In an application, one could partition a large transaction graph into subgraphs and identify anomalies on the subgraph level.

For graph-level anomaly detection, we assume to have access to a large dataset of typical graphs, such as a dataset of communities in a social network or a dataset of molecules. All graphs in the training data are considered “normal”. The goal is to use the data to learn an anomaly scoring function which can then be used to score how likely it is that a new graph is either normal or abnormal. Importantly, the term graph-level anomaly detection refers to detecting *entire* abnormal graphs, rather than localizing anomalies within graphs.

Finding abnormal nodes or edges within a large graph is widely studied [144]. In contrast, deep learning for graph-level anomaly detection has received less attention. Zhao and Akoglu [145] first explored how to extend deep OCC for graph-level anomaly detection and developed One-Class GIN (OCGIN). They also introduced two-stage graph-level anomaly detection frameworks with either graph embedding models or graph kernels. However, all these attempts have not yet produced a solid baseline for graph-level anomaly detection. Zhao and Akoglu [145] report that these methods suffer from “performance flip”, where the trained model systematically confuses anomalies with normal samples. We study how to overcome it and raise the bar in graph-level anomaly detection.

In this section, we develop One-Class Graph Transformation Learning (OCGTL) and Multi-view One-Class Classification (MOCC) for graph-level anomaly detection. Both methods combine deep OCC and self-supervision. Specifically, OCGTL uses a joint loss with a one-class learning term and a transformation learning term. MOCC is a generalization of deep OCC that involves multiple learnable views and datapoint-dependent centers. Our experiments in Section 3.2.3 reveal that the proposed methods, OCGTL and MOCC, improve over existing methods in graph-level anomaly detection significantly.

3.2.1 Algorithm and Theory of OCGTL

OCGTL combines the complementary strengths of deep OCC [89, 145] and self-supervised anomaly detection with learnable transformations (see Section 2.1). Combining these two learning paradigms is advantageous for two reasons. First, deep OCC is prone to a trivial solution called *hypersphere collapse*, which cannot be used for anomaly detection. OCGTL provably overcomes hypersphere collapse by regularizing the one-class learning term in the objective with a transformation learning term. The resulting model is more flexible (the hypersphere center can be treated as a trainable parameter) and training is more robust, despite the added flexibility. Second, the two loss contributions focus on different notions of distance between the graph embeddings. The one-class learning term relies on Euclidean distances while the transformation learning term is sensitive to the angles between embeddings. When the combined loss is used as the anomaly score this means that it can detect abnormal embedding configurations both in terms of angles between embeddings and in terms of Euclidean distances.

As follows, we first introduce OCGTL and then detail its main ingredients, including self-supervised anomaly detection with learnable transformations, deep OCC, and feature extraction with Graph Neural Networks (GNNs). We then present the theory behind OCGTL.

3.2.1.1 The OCGTL Method

OCGTL consists of an ensemble of GNNs. One of them – the reference feature extractor $\mathcal{T}_0^\theta$ – produces a reference embedding of its input graph. The other K GNN feature extractors $\mathcal{T}_k^\theta$ $(k = 1, \cdots, K)$ produce alternative "latent views" of the graph. Each of them takes as input an attributed graph $\mathrm{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{X}\}$, with vertex set $\mathcal{V}$, edges $\mathcal{E}$, and node features (attributes) $\mathcal{X} = \{\boldsymbol{x}_v | v \in \mathcal{V}\}$ and maps it into an embedding space $\mathcal{Z}$. These $K + 1$ feature extractors are trained on the training set $\mathcal{D}$ jointly with the OCGTL loss, $\mathcal{L}^\theta_{\mathrm{OCGTL}} = \mathbb{E}_{\mathrm{G}\sim\mathcal{D}}\left[\ell_{\mathrm{OCGTL}}(\mathrm{G};\theta)\right]$. Each graph in the training data contributes two terms to the loss,

$$\ell_{\mathrm{OCGTL}}(\mathrm{G};\theta) = \ell_{\mathrm{OCC}}(\mathrm{G};\theta) + \lambda \cdot \ell_{\mathrm{GTL}}(\mathrm{G};\theta). \tag{3.9}$$

The first term, $\ell_{\mathrm{OCC}}(\mathrm{G};\theta)$, is a one-class learning term; it encourages all the embeddings to be as close as possible to the same point $\boldsymbol{c} \in \mathcal{Z}$. The second term, $\ell_{\mathrm{GTL}}(\mathrm{G};\theta)$, is a Graph Transformation Learning (GTL) term; it enforces each GNN's embeddings to be diverse and semantically meaningful representations of the input graph G. λ is a balancing hyperparameter and is one in default. The tension that arises from satisfying both aspects of the objective leads to a harder self-supervision task, which in turn leads to better anomaly detection performance. When the training objective is difficult to satisfy, the trained model has to be more sensitive to typical salient features of normal data. New graphs which do not exhibit these features incur a higher loss and are then more easily detected as anomalies.

The two terms are presented in detail below.

Graph transformation learning term. We adopt DCL (Equation (2.2)) to graphs. For a graph G, the loss of GTL encourages the embeddings of each GNN, $\mathcal{T}_k^\theta(\mathrm{G})$, to be similar to the embedding of the reference GNN, $\mathcal{T}_0^\theta(\mathrm{G})$, while being dissimilar from each other. Consequently, each GNN is able to extract graph-level features to produce a different view of G. The contribution of each graph to the objective is

$$\ell_{\mathrm{GTL}}(\mathrm{G};\theta) = -\sum_{k=1}^{K} \log \frac{h(\mathcal{T}_k^\theta(\mathrm{G}), \mathcal{T}_0^\theta(\mathrm{G}))}{\sum_{l\in\{0,\ldots,K\}/\{k\}} h(\mathcal{T}_k^\theta(\mathrm{G}), \mathcal{T}_l^\theta(\mathrm{G}))} \tag{3.10}$$

One-class learning term. OCC is a popular paradigm for anomaly detection [13, 89]. The idea is to map data into a minimal hypersphere encompassing all normal training data. Data points outside the boundary are considered anomalous. The contribution of each graph G to our OCC objective is

$$\ell_{\mathrm{OCC}}(\mathrm{G};\theta) = \sum_{k=1}^{K} \|(\mathcal{T}_k^\theta(\mathrm{G}) - \boldsymbol{c})\|_2 \tag{3.11}$$

The loss function penalizes the distance of the graph G to the center $\boldsymbol{c}$ which we treat as a trainable parameter. In previous deep OCC approaches, the center $\boldsymbol{c}$ has to be a fixed hyperparameter to avoid trivial solutions to Equation (3.11).

Feature extraction with GNNs. For graph data, parametrizing the feature extractors $\mathcal{T}_0^\theta, \ldots, \mathcal{T}_K^\theta$ by GNNs is advantageous. At each layer l, a GNN maintains node representation vectors $h_v^{(l)}$ for each node v. The representation is computed based on the previous layer's representations of v and its neighbors $\mathcal{N}(v)$,

$$\boldsymbol{z}_v^{(l)} = \mathrm{GNN}^{(l)}\left(\boldsymbol{z}_v^{(l-1)}, \boldsymbol{z}_u^{(l-1)} \mid u \in \mathcal{N}(v)\right). \tag{3.12}$$

Each layer's node representations are then combined into layer-specific graph representations,

$$\boldsymbol{z}_{\mathrm{G}}^{(l)} = \mathrm{READOUT}^{(l)}\left(\boldsymbol{z}_v^{(l)} \mid v \in G\right), \tag{3.13}$$

which are concatenated into graph-level representations,

$$\boldsymbol{z} = \mathrm{CONCAT}\left(\boldsymbol{z}_{\mathrm{G}}^{(l)} \mid l = 1, \ldots, L\right). \tag{3.14}$$

This concatenation introduces information from various hierarchical levels into the graph representation [146]. Our empirical study in Section 3.2.3 shows that the choice of the readout function (which determines how the node representations are aggregated into graph representations) is particularly important to detect graph-level anomalies reliably.

Anomaly scoring with OCGTL. OCGTL is an end-to-end method for graph-level anomaly detection. During test, ℓ_{OCGTL} (Equation (3.9)) is also used directly as the score function for detecting anomalous graphs. A low loss on a test sample means that the graph is likely normal, whereas a high loss is indicative of an anomaly. One advantage of OCGTL is that its loss makes it more sensitive to different types of anomalies by considering different notions of distance between the graph embeddings. The one-class learning term is based on Euclidean distances, while the transformation learning loss is based on angles between embeddings. With the combined loss as the anomaly score, our method is sensitive to abnormal embedding configurations both in terms of angles between the latent views and in terms of Euclidean distances. In contrast, OCC-based methods typically rely on the Euclidean distance only.

Another advantage of OCGTL over OCC-based approaches is that its training is more robust and the model can be more flexible. We prove this next.

3.2.1.2 A Theory of OCGTL

A known difficulty for training OCC-based deep anomaly detectors (such as DSVDD and OCGIN) is *hypersphere collapse* [89]. Hypersphere collapse is a trivial optimum of the training objective and occurs when the model maps all inputs exactly into the center. The hypersphere then has a radius of zero, and anomaly detection becomes impossible. Ruff et al. [89] recommend fixing the center and removing bias terms for the model and show good results in practice. However, there is no guarantee that a trivial solution can be avoided under any model architecture. Here we prove that OCGTL overcomes this.

We show that the trivial solution of Equation (3.11), $\mathcal{T}_k^\theta(\mathrm{G}) = \boldsymbol{c}$ achieves a perfect OCC loss but is not optimal under the OCGTL loss. Thus OCGTL provably avoids hypersphere collapse even when the center $\boldsymbol{c}$ is a trainable parameter. This result makes OCGTL the first deep OCC approach where the center can be trained.

Proposition 8. The constant feature extractors, $\mathcal{T}_k^\theta(\mathrm{G}) = \boldsymbol{c}$ for $k = 1, \cdots, K$ and all inputs G, minimize $\ell_{\text{OCC}}(\mathrm{G};\theta)$ (Equation (3.11)).

Proof. $0 \leq \ell_{\text{OCC}}(\mathrm{G};\theta)$ is the Euclidean norm of the distance between the embedding of G and the center $\boldsymbol{c}$. Plugging in $\mathcal{T}_k^\theta(\mathrm{G}) = \boldsymbol{c}$ attains the minimum 0. □

In contrast, regularization with transformation learning can avoid hypersphere collapse. Under the constant encoder, each view is equidistant from the reference embedding and the other latent views, leading to $\ell_{\text{GTL}}(\mathrm{G};\theta) = K \log K$. However, the transformation learning objective aims at making the views predictive of the reference embeddings, in which case $\ell_{\text{GTL}}(\mathrm{G};\theta) < K \log K$. The following proposition shows that if there is a parameter setting that achieves this, the constant feature extractors do not minimize the OCGTL loss which proves that hypersphere collapse can be avoided.

Proposition 9. If there exists a parameter setting such that $\ell_{\mathrm{GTL}}(\mathrm{G};\theta) < K \log K$ on the training data, then the constant feature extractors $\mathcal{T}_k^\theta(\mathrm{G}) = \boldsymbol{c}$ do not minimize the combined loss $\ell_{\mathrm{OCGTL}}(\mathrm{G};\theta)$ (Equation (3.9)).

Proof. For constant feature extractors $\mathcal{T}_k^\theta(\mathrm{G}) = \boldsymbol{c}$ (for $k = 1, \ldots, K$) and all inputs G, the combined loss $\ell_{\mathrm{OCGTL}}(\mathrm{G};\theta) = \ell_{\mathrm{GTL}}(\mathrm{G};\theta) \geq K \log K$, where K is the number of transformations and $K \log K$ is the negative entropy of randomly guessing the reference embedding. Assume there is a constellation of the model parameters s.t. $\ell_{\mathrm{GTL}}(\mathrm{G};\theta) < K \log K$. Since $\boldsymbol{c}$ is trainable, we can set it to be the origin. The loss of the optimal solution is at least as good as the loss with $\boldsymbol{c} = 0$. Set $\epsilon = K \log K - \ell_{\mathrm{GTL}}(\mathrm{G};\theta)$. The transformations can be manipulated such that their outputs are rescaled and as a result their sum $\sum_{k=1}^{K} ||\mathcal{T}_k^\theta(\mathrm{G})||_2 < \epsilon$. As the norm of the embeddings changes, $\ell_{\mathrm{GTL}}(\mathrm{G};\theta)$ remains unchanged since the cosine similarity is not sensitive to the norm of the embeddings. By plugging this into Equation (3.9) we get $\ell_{\mathrm{OCGTL}}(\mathrm{G};\theta) = \sum_{k=1}^{K} ||\mathcal{T}_k^\theta(\mathrm{G})||_2 + \ell_{\mathrm{GTL}}(\mathrm{G};\theta) < K \log K$, which is better than the performance of the best constant encoder. □

Propositions 8 and 9 demonstrate that while deep OCC is prone to hypersphere collapse, the same trivial solution is not a minimizer of the combined loss $\ell_{\mathrm{OCGTL}}(\mathrm{G};\theta)$. The assumption of Proposition 9, that $\ell_{\mathrm{GTL}}(\mathrm{G};\theta) < K \log K$ can be tested in practice by training GTL and evaluating the predictive entropy on the training data. In all scenarios we worked with $\ell_{\mathrm{GTL}}(\mathrm{G};\theta) << K \log K$ after training.

3.2.2 Transformation Learning as Multi-View OCC

GTL can be an end-to-end graph-level anomaly detection method on its own. The $K+1$ GNNs, $\mathcal{T}_k^\theta$ $(k = 0, \cdots, K)$ are jointly trained on ℓ_{GTL} (Equation (3.10)). The loss ℓ_{GTL} is then used directly to score anomalies. However, ℓ_{GTL} is not sensitive to the embedding's norm. The norm of the graph embeddings can reflect the graph size as it is derived from aggregating the node representations. We know that graph size acts as an important feature in graph-related tasks, such as graph classification [147]. The embedding normalization step in ℓ_{GTL} (computing the cosine similarity) erases the norm information. This may put GTL at a disadvantage compared to the other methods, which profit from being aware of the graph size. To improve GTL on extracting features from the embedding norms, we develop MOCC, which incorporates OCC into transformation learning by simply modifying the similarity function of GTL. We also present that MOCC is a multi-view generalization of deep OCC and has several advantages thanks to the learnable transformations.

We first specialize ℓ_{GTL} so that it has a similar form to ℓ_{OCC}. To this end, we set the similarity function to the negative Euclidean distance. This results in the MOCC loss, a multi-view generalization of deep OCC loss involving multiple mappings $\mathcal{T}_k^\theta(\cdot) : \mathcal{X} \to \mathcal{Z}$,

$$\begin{aligned}
\ell_{\mathrm{MOCC}}(\mathrm{G};\theta) &:= -\tau \sum_{k=1}^{K} \log \frac{\exp(-||\mathcal{T}_k^\theta(\mathrm{G}) - \mathcal{T}_0^\theta(\mathrm{G})||_2/\tau)}{\sum_{l\in\{0,\ldots,K\}/\{k\}} \exp(-||\mathcal{T}_k^\theta(\mathrm{G}) - \mathcal{T}_l^\theta(\mathrm{G})||_2/\tau)} \\
&= \sum_{k=1}^{K} ||\mathcal{T}_k^\theta(\mathrm{G}) - \mathcal{T}_0^\theta(\mathrm{G})||_2 + \tau \log \mathrm{Z}_k \,, \qquad (3.15) \\
&\text{where} \quad \mathrm{Z}_k = \sum_{l\in\{0,\ldots,K\}/\{k\}} \exp(-||\mathcal{T}_k^\theta(\mathrm{G}) - \mathcal{T}_l^\theta(\mathrm{G})||_2/\tau).
\end{aligned}$$

After rearranging the loss terms, the connection to deep OCC becomes apparent. The ℓ_{MOCC} ties together K OCC-type losses and a regularization term. Instead of the fixed center $\boldsymbol{c}$, the one-class learning terms now have datapoint-dependent centers $\mathcal{T}_0^\theta(\mathrm{G})$. There are multiple embeddings $\mathcal{T}_k^\theta(\mathrm{G})$ of each graph, and the regularizer prevents all embeddings from being the same. The log-partition function $\log \mathrm{Z}_k$ pushes different views away from each other, ensuring that K transformations do not collapse to the same value in the embedding space. For $K = 1$ and removing $\log \mathrm{Z}_k$ (or sending $\tau \to 0$), ℓ_{MOCC} is equivalent to ℓ_{OCC} when $\mathcal{T}_0^\theta(\mathrm{G})$ is replaced by a constant, independent of the input graph G.

Discussion. We empirically find that MOCC is more powerful than deep OCC (see Section 3.2.3), which could be explained as follows. First, compared to deep OCC, MOCC learns to extract multiple, different features for anomaly detection (as opposed to only one feature) by including multiple learnable views. All extracted features are enforced to be diverse by the regularization term. Second, $\mathcal{T}_0^\theta(\mathrm{G})$ serves as a datapoint-dependent center. Compared with the fixed center $\boldsymbol{c}$ in ℓ_{OCC}, $\mathcal{T}_0^\theta(\mathrm{G})$ is more flexible since it can be optimized jointly with the other parameters. $\mathcal{T}_0^\theta(\mathrm{G})$ is also more informative since it preserves instance-level information beyond the common factors of variation in the dataset at large. So MOCC incorporates the one-class learning into the transformation learning and becomes more powerful and more flexible than deep OCC.

Interestingly, the NTL loss and ℓ_{MOCC} do not couple any data points explicitly, while ℓ_{OCC} couples the data points by pulling them to a shared center $\boldsymbol{c}$. NTL and MOCC seek to find "intrinsic" features within learnable views that characterize the normality of data. By measuring the similarity of each view/feature with the reference embedding $\mathcal{T}_0^\theta(\mathrm{G})$, MOCC detects the anomalies.

3.2.3 Experiments on Graph-level Anomaly Detection

This section details our empirical study. We benchmark ten algorithms on six real-world graph classification datasets from different domains using various evaluation measures. First, we describe the datasets and how the anomaly detection benchmark is set up. Second, we present the baselines and their implementation details. Third, the evaluation results are presented and analyzed. We show that our proposed methods (OCGTL and MOCC) outperform the baselines on real-world datasets from various domains and raise the anomaly detection accuracy significantly. Finally, we present our findings about preferable design choices that are also beneficial to other deep methods in graph-level anomaly detection.

3.2.3.1 Datasets and Experimental Setting

We benchmark ten methods on six graph classification datasets that are representative of three domains. In addition to financial and social network security, health organizations need an effective graph-level anomaly detection method to examine proteins (represented as graphs) to monitor the spread and evolution of diseases. Targeting these application domains, we study two bioinformatics datasets: DD and PROTEINS, two molecular datasets: NCI1 and AIDS, and two datasets of social networks: IMDB-BINARY and REDDIT-BINARY. The datasets are made available by Morris et al. [148], and the statistics of the datasets are given in Appendix B.2.

We follow the standard setting of previous work to construct an anomaly detection task from a classification dataset [38, 89, 145]. A classification dataset with $\mathcal{N}$ classes produces $\mathcal{N}$ experimental variants. In each experimental variant, one of the classes is treated as "normal"; the other classes are considered anomalies. The training set and validation set only contain normal samples, while the test set contains a mix of normal samples and anomalies that have to be detected during test time. For each experimental variant, 10% of the normal class is set aside for the test set, and 10% of each of the other classes is added to the test set as anomalies. The resulting fraction of anomalies in the test set is proportional to the class balance in the original dataset. The remaining 90% of the normal class is used for training and validation. We use 10-fold cross-validation to estimate the model performance. In each fold, 10% of the training set is held out for validation. We train each model three times separately and average the test results of three runs to get the final test results in each fold. Training multiple times ensures a fair comparison as it favors methods that are robust to random initialization.

Evaluation. Results will be reported in terms of AUC (%), averaged over ten folds with standard deviation. In addition, all methods will be evaluated in terms of their susceptibility to performance flip. Zhao and Akoglu [145] coined the term "performance flip" for anomaly detection benchmarks derived from binary classification datasets. We generalize their definition to multiple classes:

Definition 3.2.1. (Performance flip.) A model suffers from the performance flip on an anomaly detection benchmark derived from a classification dataset if it performs worse than random on at least one experimental variant.

3.2.3.2 Baselines and Implementation Details

Many deep anomaly detection approaches that achieved impressive results in other domains have not yet been adapted to graph-level anomaly detection. There has been no comprehensive study of various GNN-based graph-level anomaly detection approaches. An additional contribution of our work is that we adapt recent advances in deep anomaly detection to graphs. In our empirical study, we compare OCGTL and MOCC to GNN-based methods and to non-GNN-based methods.

GNN-based baselines. Our study includes OCGTL, MOCC, OCGIN [145], an ablation GTL, and a newly proposed self-supervised approach, Graph Transformation Prediction (GTP). GTP is an end-to-end self-supervised method based on transformation prediction. It trains a classifier to predict which transformation has been applied to a sample and uses the cross-entropy loss to score anomalies. We implement GTP with six graph transformations (node dropping, edge dropping, edge adding, attribute masking, subgraph, and identity transformation) originally designed in You et al. [149].

We use Graph Isomorphism Network (GIN) [147] as the feature extractor for all GNN-based baselines to compare with OCGIN fairly. In particular, we use four GIN layers, each of which includes a two-layer MLP and graph normalization [150]. The dimension of the node representations is 32. The readout function of almost all methods consists of a two-layer MLP, and then an add pooling layer. In GTP, the final prediction is obtained by summing the layer-wise predictions, and the readout function is composed of an add pooing layer followed by a linear layer. In GTP, we employ six hand-crafted transformations. For a fair comparison, OCGTL, MOCC, and GTL also use six learnable graph transformations in all experiments. Additional implementation details are provided in Appendix B.2.

Non-GNN-based baselines. We include four two-stage detection methods proposed by Zhao and Akoglu [145]. Two of them use unsupervised graph embedding methods, Graph2Vec (G2V) [151] or FGSD[152], to extract graph-level representations. The other two of them make use of graph kernels (Weisfeiler-Leman subtree Kernel (WLK) [153] or Propagation Kernel (PK) [154]), which measure the similarity between graphs. For all two-stage detection baselines, we use OCSVM (with $\nu = 0.1$)

Table 3.2: Average AUC (%) with standard deviations of ten methods on six datasets. Results marked with * perform worse than random on at least one experimental variant (performance flip). OCGTL and MOCC both outperform other baselines on all datasets and have no performance flip.

Datasets	DD	PROT	NCI1	AIDS	IMDB-B	RDT-B	Avg.
WLK	50.2±0.3*	49.7±0.5*	49.6±0.4*	51.3±0.8*	62.0±2.0	50.2±0.2*	52.2
PK	51.2±2.3*	50.8±1.5*	51.4±1.7*	59.5±2.3*	53.5±2.0	50.0	52.7
G2V	49.5±2.2*	53.2±3.0*	50.5±0.7*	48.4±0.8*	54.3±1.6	51.5±0.6*	51.2
FGSD	66.0±2.3	58.5±1.8*	55.4±0.7	91.6±3.7	57.1±1.8	-	65.7
OCGIN	50.7±1.2*	54.2±1.2*	53.6±1.3*	60.8±2.2*	60.4±2.8	67.1±3.5	57.8
OCPool	61.1±3.3*	**61.9**±**2.1**	57.0±1.3	97.5±1.4	56.5±0.8	65.3±2.2	66.6
GTP	54.2±1.9	**61.9**±**2.9**	55.3±1.2*	77.2±2.9	57.6±1.1	64.4±2.2	61.8
GTL	51.7±0.9*	56.2±2.5*	59.8±1.0*	67.8±3.3*	**65.2**±**1.9**	71.6±2.3	62.1
OCGTL	**69.9**±**2.6**	60.7±2.4	63.7±1.2	97.5±2.0	65.1±1.8	77.4±1.9	**72.4**
MOCC	67.0±2.0	61.8±1.5	**64.8**±**1.6**	**98.5**±**1.2**	62.2±2.2	**79.1**±**2.4**	72.2

as the downstream outlier detector. The number of iterations specifies how far neighborhood information can be propagated. By setting the number of iterations to 4, we get a fair comparison to the GNN-based methods, which all have 4 GNN layers. All other hyperparameters correspond to the choices in Zhao and Akoglu [145]. Additional details are provided in Appendix B.2.

We also develop a new two-stage method, One-Class Pooling (OCPool), a shallow method that uses pooling to construct a graph-level representation from the node features:

$$\boldsymbol{z} = \text{POOLING}\left(\boldsymbol{x}_v \mid v \in \mathrm{G}\right). \tag{3.16}$$

This feature extractor does not have parameters and hence requires no training. Anomalies can be detected by training an OCSVM [13] on these features. This novel approach for graph-level anomaly detection is a simple baseline and achieves solid results in our empirical study (even though it does not use the edge sets $\mathcal{E}$ of the graphs). Another reason for studying OCPool is that it helps us understand which pooling function might work best as a readout function (Equation (3.13)) for GNN-based anomaly detection methods.

3.2.3.3 Empirical Results

We compare OCGTL and MOCC to all existing baselines on six real-world datasets. The detection results are reported in Table 3.2 in terms of average AUC (%) with standard deviation. We can see that OCGTL achieves competitive results on all datasets and outperforms the existing baselines at least by 5.8%, averaging over six datasets. MOCC also performs competitive with OCGTL and outperforms the

Table 3.3: Average AUC (%) with standard deviations on both experimental variants of bioinformatic and molecular datasets. Each dataset leads to two experimental variants by considering either class 0 or class 1 as the normal class (and the rest classes as abnormal). The results worse than random (AUC < 50%) are colored with red. OCGTL and MOCC have no performance flip on any of the datasets.

	DD		PROT		NCI1		AIDS	
Nor. Cls	0	1	0	1	0	1	0	1
WLK	81.8±2.7	18.5±2.7	78.4±5.4	20.9±6.0	33.7±3.2	65.4±3.0	96.8±1.7	5.8±1.8
PK	73.8±3.8	28.5±3.5	72.6±5.3	29.1±5.4	47.4±3.8	55.5±1.6	87.5±1.6	31.6±4.5
G2V	32.8±4.1	66.2±5.0	45.2±7.2	61.1±5.5	67.9±2.9	33.1±2.6	2.6±1.8	94.2±2.3
FGSD	76.7±3.2	55.3±3.6	70.7±6.0	46.4±5.9	56.0±2.7	54.9±2.5	84.1±7.2	99.1±1.2
OCGIN	75.2±3.4	26.3±2.7	65.9±4.5	42.5±4.4	42.9±3.0	64.4±2.5	95.5±2.2	26.0±3.9
OCPool	77.0±4.2	45.2±5.1	70.1±6.7	53.8±4.2	57.7±3.2	56.3±2.9	96.1±2.6	99.2±1.0
GTP	54.2±3.1	54.3±1.8	70.3±4.3	53.5±4.6	46.0±3.2	64.7±3.1	93.3±2.0	61.1±5.1
GTL	79.7±2.6	23.8±2.6	75.8±5.2	36.6±6.7	44.7±2.8	74.9±2.5	97.5±1.5	38.1±6.6
OCGTL	73.0±2.9	66.8±4.6	58.1±6.1	63.2±5.4	56.2±2.5	71.2±3.0	95.7±3.6	99.3±0.9
MOCC	67.1±1.8	67.0±4.7	61.2±4.6	62.4±3.8	57.1±3.3	72.6±1.6	98.0±1.5	99.1±1.0

existing baselines at least by 5.6% averaging over six datasets. OCGTL achieves the best average performance on bioinformatic datasets and social-network datasets, while MOCC performs the best on molecular datasets. We can conclude that OCGTL and MOCC raise the detection accuracy in graph-level anomaly detection on various application domains significantly.

Moreover, methods with performance flip are marked with a * in Table 3.2. We also report the results of OCGIN [145] and our methods, OCGTL and MOCC, on both experimental variants of datasets where the performance flip is observed in Table 3.3. We can see that all existing baselines suffer from the performance flip issue, while OCGTL and MOCC have no performance flip on any of the datasets.

Ablation study of methods. Here we discuss the results in Table 3.2 from the perspective of an ablation study to understand the advantages of combing deep OCC and neural transformation learning. From the results, we can see that OCGTL and MOCC improve over OCGIN consistently. Either adding ℓ_{GTL} as a regularization term in OCGTL or generalizing deep OCC with multiple learnable transformations in MOCC boost the detection performance of deep OCC (OCGIN) significantly. Also, OCGTL and MOCC outperform GTL on 5 of 6 datasets by incorporating the idea of OCC in learning and detection. The detection in terms of Euclidean distances is sensitive to the graph size, a critical feature in graph-level anomaly detection, and therefore enhances the graph-level anomaly detection performance. We can conclude that the two learning paradigms complement each other. As a result, OCGTL and MOCC consistently outperform OCGIN and GTL.

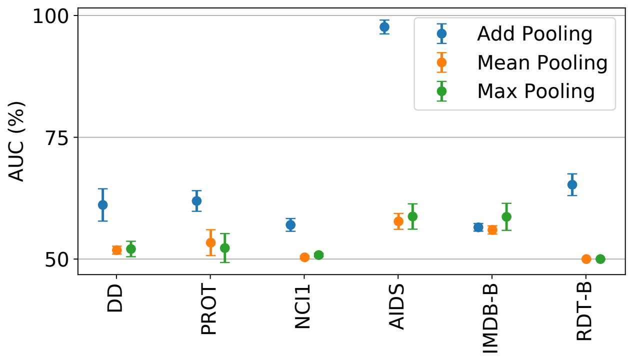


Figure 3.5: OCPool with add pooling (blue) outperforms alternative choices (mean (orange), max (green)). The results are reported in terms of AUC (%).

GTP applies hand-crafted graph transformations. Its performance varies across datasets since it is sensitive to the choice of transformations. Even though it works well on the PROTEINS dataset with the chosen graph transformations, its performance on other datasets is not competitive with OCGTL and MOCC. Finding the right transformations for each dataset requires domain knowledge, which is not the focus of this work. In comparison, OCGTL and MOCC learn data-specific transformations automatically and perform consistently well on various datasets.

Study of design choices. To raise the bar in deep anomaly detection on graphs, we must understand the impact of the design choices associated with the GNN architecture. Here we discuss the type of pooling layer for the readout function (Equation (3.13)) and the normalization of the GNN layers.

First, we study the impact of different pooling layers. OCPool is the ideal testbed to compare add pooling, mean pooling, and max pooling due to its simplicity. Results of running the three options on nine datasets are reported in Figure 3.5. Add pooling outperforms the other options. This may result from that add pooling injects information about the number of nodes into the graph representations. OCPool with add pooling is a simple yet effective method for anomaly detection. It provides a simple heuristic for aggregating node attributes into graph representations. As shown in Table 3.2, it does well on many datasets (particularly on the PROTEINS and AIDS datasets), even though it does not account for graph structure (edges).

Second, to study the combined impact of the pooling layer and normalization layers on the *deep* GNN-based methods, we compare add pooling with graph normalization (AP + GN) and mean pooling with batch normalization (MP + BN). In Figure 3.6, we visualize in a scatter plot the performance of all GNN-based methods on all six datasets, contrasting the AP + GN result with the MP + BN result in terms of average AUC (%) with standard deviation. Almost all points fall above the diagonal,

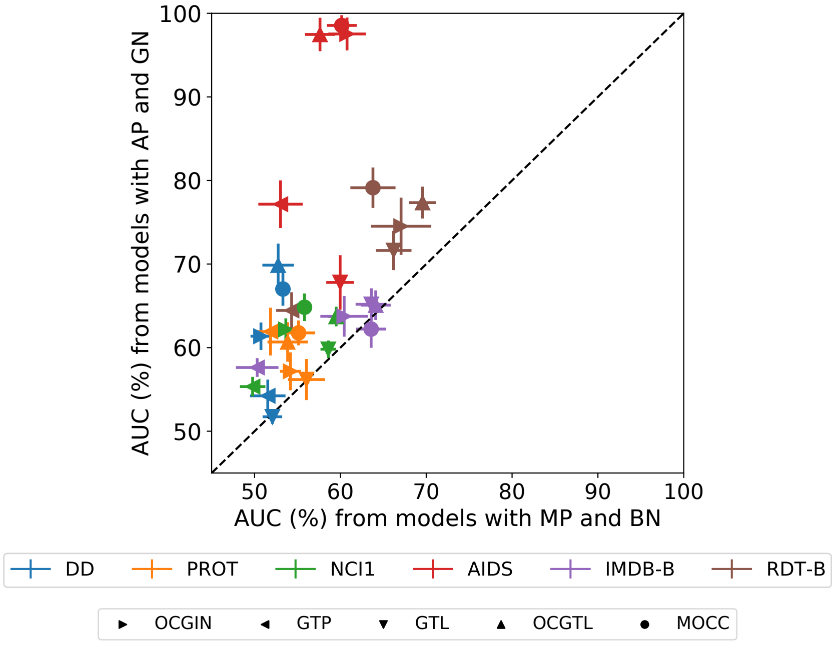


Figure 3.6: A comparison of two design choices in *deep* GNN-based methods, namely, add pooling with graph normalization (AP + GN, y-axis) and mean pooling with batch normalization (MP + BN, x-axis). Each point compares the detection results of the two variants of one model on one dataset. Most points falling above the diagonal indicate that AP + GN is the preferred design choice for deep GNN-based anomaly detection methods.

meaning that AP + GN is preferable to MP + BN, which has been the design choice of Zhao and Akoglu [145] for OCGIN. With the new design choices, we are able to significantly raise the bar in graph-level anomaly detection.

3.3 Summary and Discussion

For detecting anomalies within time series, we propose a novel self-supervised method LNT. The key ingredient is a new training objective combining representation learning and transformation learning. We prove that both learning paradigms complement each other to avoid trivial solutions not appropriate for anomaly detection within time series. We find in an empirical study that LNT learns diverse and semantically meaningful transformations, leading to the improvement over many strong baselines on challenging detection tasks.

For graph-level anomaly detection, we develop end-to-end methods, OCGTL and MOCC, that combine NTL and deep OCC. OCGTL mitigates the shortcomings of deep OCC by using graph transformation learning as regularization and complements graph transformation learning by introducing a sensitivity to the norm of the graph representations. MOCC generalizes deep OCC and becomes a more powerful and more flexible anomaly detection method thanks to the learnable transformations. Our comprehensive empirical study supports our claim and theoretical results. It shows that our proposed methods (OCGTL and MOCC) outperform existing work in various challenging domains and do not struggle with performance flip.

Parts of this chapter are mainly based on:

[155] Tim Schneider, **Chen Qiu**, and Maja Rudolph. Anomalous Region Detection in Time Series with Local Neural Transformations. In *ICML 2021 Workshop: Self-Supervised Learning for Reasoning and Perception*, 2021.

[156] **Chen Qiu**, Marius Kloft, Stephan Mandt, and Maja Rudolph. Raising the Bar in Graph-level Anomaly Detection. In *In Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, IJCAI-22, pages 2196–2203. International Joint Conferences on Artificial Intelligence Organization, 7 2022. Main Track.

Chen Qiu, Marius Kloft, Stephan Mandt, and Maja Rudolph. Self-Supervised Anomaly Detection with Neural Transformations. *Preprint (under review)*, 2022.

4 Neural Transformation Learning with Contaminated Data

A common assumption in training anomaly detection models is that clean training data is available to teach the model what normal samples look like [17]. In reality, this assumption is often violated: datasets are frequently large and uncurated and may already contain some of the anomalies one is hoping to find. For example, a large dataset of medical images may already contain cancer images, or datasets of financial transactions could already contain unnoticed fraudulent activity. In this chapter, we study how to learn NTL on a contaminated dataset.

In Section 4.1, we propose a strategy, Latent Outlier Exposure (LOE), for training deep anomaly detectors in the presence of unlabeled anomalies. LOE is compatible with a broad class of models, including NTL, and improves their robustness to data contamination. The idea is to jointly infer binary labels to each datum (normal vs. anomalous) while updating the model parameters. We use a combination of two losses that share parameters: one for the normal and one for the anomalous data. We then iteratively proceed with block coordinate updates on the parameters and the most likely (latent) labels. Our experimental evaluation demonstrates that LOE is applicable to a variety of deep anomaly detection models and improves over the existing training strategies on the contaminated data significantly.

For many anomaly detection problems, e.g., in biomedical image analysis or fraud detection, one can achieve significantly higher performance by querying anomaly labels. In Section 4.2, we propose a new active learning strategy, Active Latent Outlier Exposure (ALOE), for deep anomaly detection. ALOE queries data diversely in feature space based on the seeding algorithm of K-means++ . By just querying a few diverse samples, ALOE can efficiently estimate the sole hyperparameter in LOE, the assumed anomaly rate, based on an importance sampling estimate. The resulting approach is hyperparameter-free and applicable to various deep anomaly detection methods. Our extensive experiments on image and tabular data show that ALOE results in state-of-the-art active anomaly detection performance.

4.1 Latent Outlier Exposure

A common strategy to deal with contaminated training data is to hope that the contamination ratio is low and that the anomaly detection method will exercise inlier priority [40]. Throughout this thesis, we refer to the strategy of blindly training an anomaly detector as if the training data was clean as "*Blind*" training. Yoon et al. [157] have proposed a data refinement strategy that removes potential anomalies from the training data. Their approach, which we refer to as "*Refine*", employs an ensemble of one-class classifiers to iteratively weed out anomalies and then continue training on the refined dataset. Similar data refinement strategies are also combined with latent SVDD [158], autoencoders [159, 160], or time series models [161] for anomaly detection. However, these methods fail to exploit the insight of outlier exposure [31] that anomalies provide a valuable training signal.

We introduce LOE, a new unsupervised approach to training deep anomaly detectors on a contaminated dataset. LOE uses a combination of two coupled losses to extract learning signals from both normal and anomalous data. In order to decide which of the two loss functions to activate for a given datum (normal vs. abnormal), we use a binary latent variable that we jointly infer while updating the model parameters. Training the model thus results in a joint optimization problem over continuous model parameters and binary variables that we solve using alternating updates. During testing, we can use threshold only one of the two loss functions to identify anomalies in constant time. We demonstrate that LOE can be applied to a variety of anomaly detection methods, including NTL (Section 2.1), and data types, including tabular, image, and video data. Beyond detection of entire anomalous images, we also consider the problem of anomaly segmentation which is concerned with finding anomalous regions within an image.

4.1.1 Algorithm and Methodology of LOE

We study the problem of unsupervised (or self-supervised) anomaly detection. We consider a dataset of samples $\boldsymbol{x}_i$; these could either come from a data distribution of "normal" samples or could otherwise come from an unknown corruption process and thus be considered "anomalies". For each datum $\boldsymbol{x}_i$, let $\tilde{y}_i = 0$ if the datum is normal, and $\tilde{y}_i = 1$ if it is anomalous. We assume that these binary labels are unobserved in our training and have to be inferred from the data.

In contrast to most anomaly detection setups, we assume that our dataset is *corrupted by anomalies*. That means we assume that a fraction $(1 - \alpha)$ of the data is normal, while its complementary fraction α is anomalous. This corresponds to a more challenging (but arguably more realistic) anomaly detection setup since the training data cannot be assumed to be normal. We treat the assumed anomaly rate α as a hyperparameter in our approach and denote α_0 as the ground truth contamination rate where needed. Note that an assumed contamination ratio is a common hyperparameter in many robust algorithms [e.g., 162, 163], and we test the robustness of our approach w.r.t. this parameter in Section 4.1.2.

Our goal is to train a (deep) anomaly detection model on such corrupted data based on self-supervised or unsupervised training paradigms. The challenge thereby is to simultaneously infer the binary labels $\tilde{y}_i$ during training while optimally exploiting this information for training the model.

4.1.1.1 Loss Function

We consider two losses in LOE. Similar to most work on deep anomaly detection, we consider a loss function $\ell_n(\boldsymbol{x};\theta)$ that we aim to minimize over "normal" data. When being trained on only normal data, the trained loss will yield lower values for normal than for anomalous data so that it can be used to construct an anomaly score.

We also consider a second loss for anomalies $\ell_a(\boldsymbol{x};\theta)$ (the parameters θ are shared). Minimizing this loss on only anomalous data will result in low loss values for anomalies and larger values for normal data. The anomaly loss is designed to have opposite effects as the loss function $\ell_n(\boldsymbol{x};\theta)$. For example, if $\ell_n(\boldsymbol{x};\theta) = ||f^\theta(\boldsymbol{x}) - \boldsymbol{c}||^2$ as in DSVDD [89] (pulling normal data towards the center), we define $\ell_a(\boldsymbol{x};\theta) = 1/||f^\theta(\boldsymbol{x}) - \boldsymbol{c}||^2$ (pushing abnormal data away from it) as in Ruff et al. [22].

Temporarily assuming that all assignment variables $\tilde{\boldsymbol{y}}$ were known, consider the joint loss function,

$$\mathcal{L}^\theta_{\text{LOE}}(\tilde{\boldsymbol{y}}) = \frac{1}{N}\sum_{i=1}^{N}(1-\tilde{y}_i)\ell_n(\boldsymbol{x}_i;\theta) + \tilde{y}_i\ell_a(\boldsymbol{x}_i;\theta). \tag{4.1}$$

This equation resembles the log-likelihood of a probabilistic mixture model, but note that $\ell_n(\boldsymbol{x}_i;\theta)$ and $\ell_a(\boldsymbol{x}_i;\theta)$ are not necessarily data log-likelihoods; rather, self-supervised auxiliary losses can be used and often perform better in practice [164].

Optimizing Equation (4.1) over its parameters θ yields a better anomaly detector than $\ell_n(\boldsymbol{x};\theta)$ trained in isolation. By construction of the anomaly loss $\ell_a(\boldsymbol{x};\theta)$, the known anomalies provide an additional training signal to $\ell_n(\boldsymbol{x};\theta)$: due to parameter sharing, the labeled anomalies teach $\ell_n(\boldsymbol{x};\theta)$ where *not* to expect normal data in feature space. This is the basic idea of Outlier Exposure [31], which constructs artificial *labeled* anomalies for enhanced detection performance.

Different from outlier exposure, we assume that the set of $\tilde{y}_i$ is unobserved, hence *latent*. We therefore term our approach of jointly inferring the latent assignment variables $\tilde{\boldsymbol{y}}$ and learning the parameters θ as *Latent Outlier Exposure*. We show that it leads to competitive performance on training data corrupted by outliers.

4.1.1.2 Optimization Problem

We discuss the optimization problem of LOE and an efficient solving procedure of the optimization problem in the following.

"Hard" Latent Outlier Exposure (LOE$_H$). In LOE, we seek to both optimize both losses' shared parameters θ while also optimizing the most likely assignment variables $\tilde{y}_i$. Due to our assumption of having a fixed rate of anomalies α in the training data, we introduce a constrained set:

$$\mathcal{Y} = \{\tilde{\boldsymbol{y}} \in \{0,1\}^N : \sum_{i=1}^{N} \tilde{y}_i = \alpha N\}. \tag{4.2}$$

The set describes a "hard" label assignment; hence the name "hard LOE", which is the default version of our approach. Section 4.1.1.3 describes an extension with "soft" label assignments. Note that we require αN to be an integer.

Since our goal is to use the losses $\ell_n(\boldsymbol{x};\theta)$ and $\ell_a(\boldsymbol{x};\theta)$ to identify and score anomalies, we seek $\ell_n(\boldsymbol{x};\theta) - \ell_a(\boldsymbol{x};\theta)$ to be large for anomalies, and $\ell_a(\boldsymbol{x};\theta) - \ell_n(\boldsymbol{x};\theta)$ to be large for normal data. Assuming these losses to be optimized over θ, our best guess to identify anomalies is to minimize Equation (4.1) over the assignment variables $\boldsymbol{y}$. Combining this with the constraint (Equation (4.2)) yields the following minimization problem:

$$\min_{\theta} \min_{\tilde{\boldsymbol{y}} \in \mathcal{Y}} \mathcal{L}^{\theta}_{\text{LOE}}(\tilde{\boldsymbol{y}}). \tag{4.3}$$

As follows, we describe an efficient optimization procedure for the constraint optimization problem.

Block coordinate descent. The constraint discrete optimization problem has an elegant solution. To this end, we consider a sequence of parameters θ^t and labels $\tilde{\boldsymbol{y}}^t$ and proceed with alternating updates. To update θ, we simply fix $\tilde{\boldsymbol{y}}^t$ and minimize $\mathcal{L}^{\theta}_{\text{LOE}}(\tilde{\boldsymbol{y}}^t)$ over θ. In practice, we perform a single gradient step (or stochastic gradient step, see below), yielding a partial update.

To update $\tilde{\boldsymbol{y}}$ given θ^t, we minimize the same function subject to the constraint (Equation (4.2)). To this end, we define training anomaly scores,

$$s^{train}(\boldsymbol{x}_i) = \ell_n(\boldsymbol{x}_i;\theta) - \ell_a(\boldsymbol{x}_i;\theta). \tag{4.4}$$

These scores quantify the effect of $\tilde{y}_i$ on minimizing Equation (4.1). We rank these scores and assign the $(1-\alpha)$-quantile of the associated labels $\tilde{y}_i$ to the value 0 and the remainder to the value 1. This minimizes the loss function subject to the label constraint. We discuss the sensitivity of our approach to the assumed rate of

Algorithm 2: Training process of LOE

```
Input: Contaminated training dataset $\mathcal{D}$ (with an anomaly rate $\alpha_0$)
       hyperparamter $\alpha$
Model: Deep anomaly detector with parameters $\theta$
// Training process
foreach Epoch do
    foreach Mini-batch $\mathcal{M}$ do
        Calculate the anomaly score $s^{train}(\boldsymbol{x}_i)$ for $\boldsymbol{x}_i \in \mathcal{M}$
        Estimate the label $\tilde{y}_i$ given $s^{train}(\boldsymbol{x}_i)$ and $\alpha$
        Update the parameters $\theta$ by minimizing $\mathcal{L}^{\theta}_{\mathrm{LOE}}(\tilde{\boldsymbol{y}})$
    end
end
```

anomalies α in our experiments section. We stress that our testing anomaly scores will be different (see Section 4.1.1.3).

Assuming that all involved losses are bounded from below, the block coordinate descent converges to a local optimum since every update improves the loss.

Stochastic optimization. In practice, we perform stochastic gradient descent on Equation (4.1) based on mini-batches. For simplicity and memory-efficiency, we impose the label constraint Equation (4.2) on each mini-batch and optimize θ and $\tilde{\boldsymbol{y}}$ in the same alternating fashion. The induced bias vanishes for large mini-batches. In practice, we found that this approach leads to satisfying results. Note that an exact mini-batch version of the optimization problem in Equation (4.3) would also be possible, requiring memorization of $\tilde{\boldsymbol{y}}$ for the whole data set.

Algorithm 2 summarizes our approach.

4.1.1.3 Model Extension and Anomaly Detection

We discuss an important extension of our approach and then present the usage of our approach in anomaly detection in the following.

"Soft" Latent Outlier Exposure (LOE$_S$). In practice, the block coordinate descent procedure can be overconfident in assigning $\tilde{\boldsymbol{y}}$, leading to suboptimal training. To overcome this problem, we also propose a *soft* anomaly scoring approach that we term *Soft* LOE. Soft LOE is very simply implemented by a modified constraint set:

$$\mathcal{Y}' = \{\tilde{\boldsymbol{y}} \in \{0, 0.5\}^N : \sum_{i=1}^{N} \tilde{y}_i = 0.5\alpha N\}. \tag{4.5}$$

Everything else about the model's training and testing scheme remains the same.

The consequence of an identified anomaly $\tilde{y}_i = 0.5$ is that we minimize an equal combination of both losses, $0.5(\ell_n(\boldsymbol{x}_i;\theta) + \ell_a(\boldsymbol{x}_i;\theta))$. The interpretation is that the algorithm is uncertain about whether to treat $\boldsymbol{x}_i$ as a normal or anomalous data point and compromises between both cases. A similar weighting scheme has been proposed for supervised learning in the presence of unlabeled examples [165]. In practice, we found the soft scheme to sometimes outperform the hard one (see Section 4.1.2).

Anomaly detection. In order to use our approach for finding anomalies in a test set, we could in principle proceed as we did during training and infer the most likely labels as described in Section 4.1.1.2. However, in practice, we may not want to assume to encounter the same kinds of anomalies that we encountered during training. Hence, we refrain from using $\ell_a(x;\theta)$ during testing and score anomalies using only $\ell_n(x;\theta)$. Note that due to parameter sharing, training $\ell_a(x;\theta)$ jointly with $\ell_n(x;\theta)$ has already led to the desired information transfer between both losses.

Testing is the same for both "soft" LOE and "hard" LOE. We define our testing anomaly score in terms of the "normal" loss function,

$$s^{test}(\boldsymbol{x}) = \ell_n(\boldsymbol{x}, \theta). \tag{4.6}$$

4.1.1.4 Neural Transformation Learning with LOE

LOE is applicable to various deep anomaly detection methods. We show the application of LOE to NTL as an example [1]. NTL learns $K+1$ neural transformations $\mathcal{T}^{\theta}_{0,\dots,K}$ with parameters θ from data and uses the learned transformations to detect anomalies. For normal samples, NTL encourages each transformed view to be similar to the original sample and to be dissimilar from other transformed views in the embedding space. To achieve this objective, NTL minimizes

$$\ell_n(\boldsymbol{x};\theta) := -\sum_{k=1}^{K} \log p_k(\boldsymbol{x};\theta) \quad \text{with} \quad p_k(\boldsymbol{x};\theta) = \frac{h(\mathcal{T}^{\theta}_k(\boldsymbol{x}), \mathcal{T}^{\theta}_0(\boldsymbol{x}))}{\sum_{l\in\{0,\dots,K\}/\{k\}} h(\mathcal{T}^{\theta}_k(\boldsymbol{x}), \mathcal{T}^{\theta}_l(\boldsymbol{x}))}, \tag{4.7}$$

where the function h defined in Equation (2.1) measures the similarity of two embeddings. Intuitively, the nominator of $p_k(\boldsymbol{x};\theta)$ pulls the transformed embeddings close to the reference embedding while the denominator pushes all transformed embeddings away from each other.

For anomalies, we "flip" the normal loss and design the abnormal loss as

$$\ell_a(\boldsymbol{x};\theta) := -\sum_{k=1}^{K} \log(1 - p_k(\boldsymbol{x};\theta)). \tag{4.8}$$

[1]Additional applications of LOE are provided in Appendix A.4.

The model instead pushes the transformed embeddings $\mathcal{T}_k^\theta(\boldsymbol{x})$ for $k = 1, \ldots, K$ away from the reference embedding $\mathcal{T}_0^\theta(\boldsymbol{x})$ and pulls them close to each other.

4.1.2 Experiments of LOE

We have conducted extensive experiments for unsupervised anomaly detection tasks on synthetic data, images, tabular data, and videos. The data are contaminated with different anomaly ratios. Our proposed method sets a new state-of-the-art on most datasets. We also show that our method gives robust results even when the contamination ratio is unknown.

Across all experiments, we employ two baselines that do not utilize anomalies to help the training of models. The baselines are either completely blind to anomalies or drop the perceived anomalies' information. Normally training a model without recognizing anomalies serves as our first baseline. Since this baseline doesn't take any action to the anomalies in the contaminated training data and is actually blind to the anomalies that exist, we name it *Blind*. Mathematically, Blind sets $\tilde{y}_i = 0$ in Equation (4.1) for all samples. The second baseline filters out anomalies and refines the training data: at every mini-batch update, it first ranks the mini-batch data according to the anomaly scores given the current detection model, then removes the top α most likely anomalous samples from the mini-batch. The remaining samples perform the model update. We name the second baseline *Refine*, which still follows Algorithm. 2 but removes $\ell_a(\boldsymbol{x}; \theta)$ in Equation (4.1). Both these two baselines take limited actions to the anomalies. We use them to contrast our proposed methods and highlight the useful information contained in unseen anomalies.

4.1.2.1 Experiments on Synthetic Data

We first analyze the methods in a controlled setup on a synthetic data set. For the sake of visualization, we created a 2D contaminated data set with a three-component Gaussian mixture. One larger component is used to generate normal samples, while the two smaller components are used to generate the anomalies contaminating the data (see Figure 4.1). For simplicity, the backbone anomaly detector is DSVDD [89] with radial basis functions. Setting the contamination ratio to $\alpha = \alpha_0 = 0.1$, we compare the baselines "Blind" and "Refine" with the proposed LOE$_H$ and LOE$_S$ (described in Section 4.1.1) and the theoretically optimal *G-truth* method which uses the ground truth labels.

Figure 4.1 shows the results (anomaly-score contour lines after training). With more latent anomaly information exploited from (a) to (e), the contour lines become increasingly accurate. While (a) "Blind" erroneously treats all anomalies as normal, (b) "Refine" improves by filtering out some anomalies. (c) LOE$_S$ and (d) LOE$_H$ use the anomalies, resulting in a clear separation of anomalies and normal data. LOE$_H$

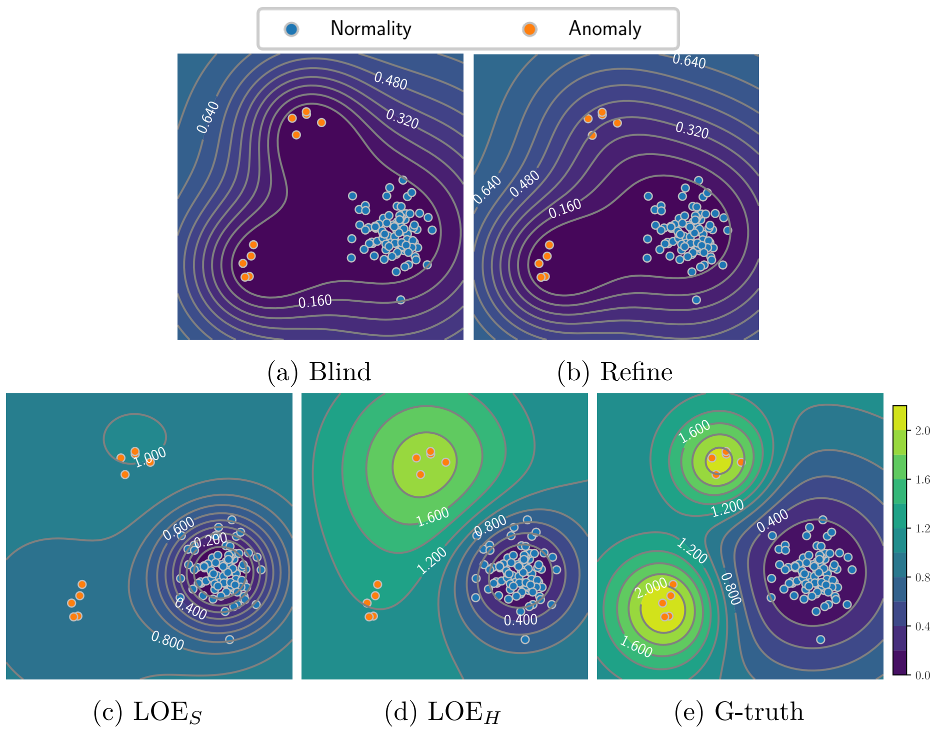


Figure 4.1: DSVDD trained on 2D synthetic contaminated data with different strategies: **(a)** "Blind" (treats all data as normal), **(b)** "Refine" (filters out some anomalies), **(c)** LOE$_S$ (proposed, assigns soft labels to anomalies), **(d)** LOE$_H$ (proposed, assigns hard labels), **(e)** supervised anomaly detection with ground truth labels (for reference). LOE leads to improved region boundaries.

leads to more pronounced boundaries than LOE$_S$, but it is at risk of overfitting, especially when normal samples are incorrectly detected as anomalies (see our experiments below). A supervised model with ground-truth labels ("G-truth") approximately recovers the true contours.

4.1.2.2 Experiments on Image Data

Anomaly detection on images is especially far developed. We demonstrate LOE's benefits when applied to NTL. Our experiments are designed to test the hypothesis that LOE can mitigate the performance drop caused by training on contaminated image data. We experiment with three image datasets: CIFAR10, FashionMNIST, and MVTEC [107]. These have been used in virtually all deep anomaly detection papers published at top-tier venues [38, 39, 41, 88, 89], and we adopt these papers' experimental protocol here, as detailed below.

Backbone model and baselines. Many existing baselines apply either blind updates or a refinement strategy to specific backbone models. However, a recent study showed that many of the classical anomaly detection methods, such as autoencoders, are no longer on par with modern self-supervised approaches [39]. By default, we use NTL as the backbone model. Results with other backbone models are shown in Appendix A.4.

NTL is built upon the final pooling layer of a pre-trained ResNet152 for CIFAR10 and FMNIST (as suggested in Defard et al. [111]), and upon the third residual block of a pre-trained WideResNet50 for MVTEC (as suggested in Reiss et al. [30]). On all image datasets, the pre-trained feature extractors are frozen during training. We set the number of transformations $K = 15$. We use a MLP of three linear layers with intermediate batchnorm layers and ReLU activations for each transformation network. The hidden sizes of the transformation networks are $[2048, 2048, 2048]$ on CIFAR10 and FMNIST, and $[1024, 1024, 1024]$ on MVTEC. The encoder is one linear layer with units of 256 for CIFAR10 and MVTEC, and is a MLP of two linear layers of size $[1024, 256]$ with an intermediate ReLU activation for FMNIST. Further details are provided in Appendix B.3.

For a more competitive and unified comparison with existing baselines in terms of the training strategy, we hence adopt the two proposed LOE methods (Section 4.1.1) and the two baseline methods “Blind” and “Refine” to NTL. For the “Refine” baseline and our methods, we set the number of warm-up epochs as two on all image datasets. In warm-up epochs, the model is trained blindly.

Image datasets. On CIFAR10 and FMNIST, we follow the standard “one-vs.-rest” protocol of converting these data into anomaly detection datasets [38, 39, 41, 89]. We create $\mathcal{N}$ anomaly detection tasks (where $\mathcal{N}$ is the number of classes), with each task considering one of the classes as normal and the union of all other classes as abnormal. For each task, the training set is a mixture of normal samples and a fraction of α_0 abnormal samples.

For MVTEC, we use patch-level features as the model inputs. The features are obtained from the third residual block of a WideResNet50 pre-trained on ImageNet as suggested in Reiss et al. [30]. Since the MVTEC training set contains no anomalies, we contaminate it with artificial anomalies that we create by adding zero-mean Gaussian noise to the features of test set anomalies. We use a large variance for the additive noise (equal to the empirical variance of the anomalous features) to reduce information leakage from the test set into the training set.

Empirical results. We present the experimental results of CIFAR10 and FMNIST in Table 4.1, where we set the contamination ratio $\alpha = \alpha_0 = 0.1$. The results are reported as the mean and standard deviation of three runs with different model initialization

Table 4.1: AUC (%) with standard deviation for anomaly detection on CIFAR10 and FMNIST. For all experiments, we set the contamination ratio as 10%. LOE mitigates the performance drop when NTL is trained on the contaminated datasets.

	CIFAR10	FMNIST
Blind	91.3±0.1 (-4.4)	85.0±0.2 (-9.7)
Refine	93.5±0.1 (-2.2)	89.1±0.2 (-5.6)
LOE$_H$ (ours)	**94.9±0.2 (-0.8)**	**92.9±0.7 (-1.8)**
LOE$_S$ (ours)	**94.9±0.1 (-0.8)**	92.5±0.1 (-2.2)

Table 4.2: AUC (%) with standard deviation for anomaly detection/segmentation on MVTEC. We set the contamination ratio of the training set as 10% and 20%. LOE mitigates the performance drop in both anomaly detection and anomaly segmentation.

	Detection		Segmentation	
	10%	20%	10%	20%
Blind	94.2±0.5 (-3.2)	89.4±0.3 (-8.0)	96.17±0.08 (-0.78)	95.09±0.17 (-1.86)
Refine	95.3±0.5 (-2.1)	93.2±0.3 (-4.2)	96.55±0.04 (-0.40)	96.09±0.06 (-0.86)
LOE$_H$	**95.9±0.9 (-1.5)**	92.9±0.4 (-4.5)	95.97±0.22 (-0.98)	93.29±0.21 (-3.66)
LOE$_S$	95.4±0.5 (-2.0)	**93.6±0.3 (-3.8)**	**96.56±0.04 (-0.39)**	**96.11±0.05 (-0.84)**

and anomaly samples for the contamination. The number in the brackets is the average performance difference to the model trained on clean data. Our proposed methods consistently outperform the baselines and mitigate the performance drop compared to the model trained on clean data. Specifically, LOE significantly improves over the best-performing baseline, "Refine", by 1.4% and 3.8% AUC on CIFAR10 and FMNIST, respectively. On CIFAR10, our methods have only 0.8% AUC lower than when training on the normal dataset.

We also evaluate our methods at various contamination ratios (from 5% to 20%) in Figure 4.2 (a) and (b). We can see 1) adding labeled anomalies (G-truth) boosts performance, and 2) among all methods that do not have ground truth labels, the proposed LOE methods achieve the best performance consistently at all contamination ratios.

We also evaluate our methods for anomaly detection and segmentation on the MVTEC dataset. Results are shown in Table 4.2, where we evaluated the methods on two contamination ratios (10% and 20%). Our method improves over the "Blind" and "Refine" baselines in all experimental settings.

4.1.2.3 Experiments on Tabular Data

Tabular data is another important application area of anomaly detection. Many datasets in the healthcare and cyber security domains are tabular data. Our empirical

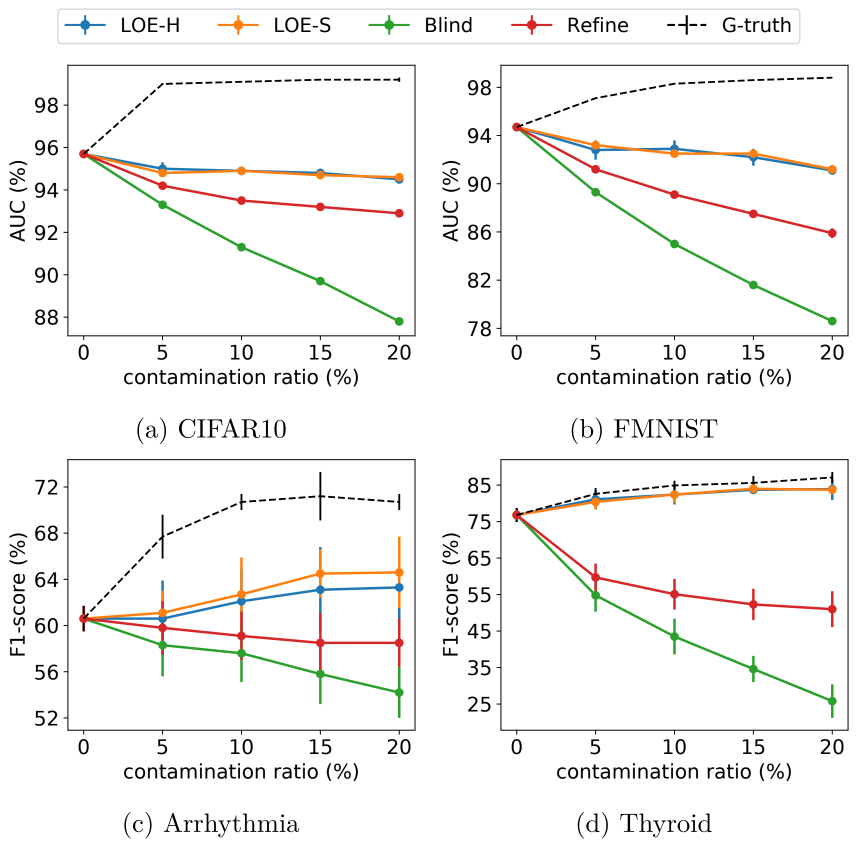


Figure 4.2: Anomaly detection performance of NTL on CIFAR10, FMNIST, and two tabular datasets (Arrhythmia and Thyroid) with $\alpha_0 \in \{5\%, 10\%, 15\%, 20\%\}$. LOE consistently outperforms the "Blind" and "Refine" on various contamination ratios.

study demonstrates that LOE yields the best performance for two popular backbone models on a comprehensive set of contaminated tabular datasets.

Tabular datasets. We study all 30 tabular datasets used in the empirical analysis of a recent state-of-the-art paper [166]. These include the frequently-studied small-scale Arrhythmia and Thyroid medical datasets, the large-scale cyber intrusion detection datasets KDD and KDDRev, and multi-dimensional point datasets from the outlier detection datasets[2]. We follow the pre-processing and train-test split of the datasets in Shenkar and Wolf [166]. To corrupt the training set, we create artificial anomalies by adding zero-mean Gaussian noise to anomalies from the test set. We use a large variance for the additive noise (equal to the empirical variance of the anomalies in the test set) to reduce information leakage from the test set into the training set.

[2]`http://odds.cs.stonybrook.edu/`

Table 4.3: F1-score (%) of NTL for anomaly detection on 30 tabular datasets studied in Shenkar and Wolf [166]. We set $\alpha = \alpha_0 = 10\%$ in all experiments. The number in the bracket is the average performance difference from the model trained on clean data. LOE outperforms the "Blind" and "Refine" baselines consistently.

	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)
abalone	37.9±13.4 (-25.3)	55.2±15.9 (-8.0)	42.8±26.9 (-20.4)	**59.3±12.0 (-3.9)**
annthyroid	29.7±3.5 (-21.6)	42.7±7.1 (-8.6)	47.7±11.4 (-3.6)	**50.3±4.5 (-1.0)**
arrhythmia	57.6±2.5 (-3.0)	59.1±2.1 (-1.5)	62.1±2.8 (+1.5)	**62.7±3.3 (+2.1)**
breastw	84.0±1.8 (-8.4)	93.1±0.9 (+0.7)	**95.6±0.4 (+3.2)**	95.3±0.4 (+2.9)
cardio	21.8±4.9 (-35.0)	45.2±7.9 (-11.6)	**73.0±7.9 (+16.2)**	57.8±5.5 (+1.0)
ecoli	0.0±0.0 (-95.6)	88.9±14.1 (-6.7)	**100±0.0 (+4.4)**	**100±0.0 (+4.4)**
forest	20.4±4.0 (-44.2)	56.2±4.9 (-8.4)	61.1±34.9 (-3.5)	**67.6±30.6 (+3.0)**
glass	11.1±7.0 (-6.7)	15.6±5.4 (-2.2)	17.8±5.4 (+0.0)	**20.0±8.3 (+2.2)**
ionosphere	89.0±1.5 (-3.5)	91.0±2.0 (-1.5)	91.0±1.7 (-1.5)	**91.3±2.2 (-1.2)**
kdd	95.9±0.0 (-2.4)	96.0±1.1 (-2.3)	98.1±0.4 (-0.2)	**98.4±0.1 (+0.1)**
kddrev	98.4±0.1 (+0.2)	98.4±0.2 (+0.2)	89.1±1.7 (-9.1)	**98.6±0.0 (+0.4)**
letter	36.4±3.6 (-11.0)	44.4±3.1 (-3.0)	25.4±10.0 (-22.0)	**45.6±10.6 (-1.8)**
lympho	53.3±12.5 (-20.0)	60.0±8.2 (-13.3)	60.0±13.3 (-13.3)	**73.3±22.6 (+0.0)**
mammo.	5.5±2.8 (-21.3)	2.6±1.7 (-24.2)	3.3±1.6 (-23.5)	**13.5±3.8 (-13.3)**
mnist	78.6±0.5 (-6.6)	**80.3±1.1 (-4.9)**	71.8±1.8 (-13.4)	76.3±2.1 (-8.9)
mulcross	45.5±9.6 (-50.5)	**58.2±3.5 (-37.8)**	58.2±6.2 (-37.8)	50.1±8.9 (-45.9)
musk	21.0±3.3 (-79.0)	98.8±0.4 (-1.2)	**100±0.0 (+0.0)**	**100±0.0 (+0.0)**
optdigits	0.2±0.3 (-24.7)	1.5±0.3 (-23.4)	41.7±45.9 (+16.8)	**59.1±48.2 (+34.2)**
pendigits	5.0±2.5 (-56.3)	32.6±10.0 (-28.7)	79.4±4.7 (+18.1)	**81.9±4.3 (+20.6)**
pima	60.3±2.6 (-1.2)	61.0±1.9 (-0.5)	**61.3±2.4 (-0.2)**	61.0±0.9 (-0.5)
satellite	73.6±0.4 (-1.0)	74.1±0.3 (-0.5)	**74.8±0.4 (+0.2)**	74.7±0.1 (+0.1)
satimage	26.8±1.5 (-65.2)	86.8±4.0 (-5.2)	90.7±1.1 (-1.3)	**91.0±0.7 (-1.0)**
seismic	11.9±1.8 (-0.6)	11.5±1.0 (-1.0)	**18.1±0.7 (+5.6)**	17.1±0.6 (+4.6)
shuttle	97.0±0.3 (+0.3)	97.0±0.2 (+0.3)	**97.1±0.2 (+0.4)**	97.0±0.2 (+0.3)
speech	6.9±1.2 (-2.6)	8.2±2.1 (-1.3)	43.3±5.6 (+33.8)	**50.8±2.5 (+41.3)**
thyroid	43.4±5.5 (-34.4)	55.1±4.2 (-22.7)	**82.4±2.7 (+4.6)**	82.4±2.3 (+4.6)
vertebral	22.0±4.5 (-8.7)	21.3±4.5 (-9.4)	22.7±11.0 (-8.0)	**25.3±4.0 (-5.4)**
vowels	36.0±1.8 (-40.7)	50.4±8.8 (-26.3)	**62.8±9.5 (-13.9)**	48.4±6.6 (-28.3)
wbc	25.7±12.3 (-39.1)	45.7±15.5 (-19.1)	**76.2±6.0 (+11.4)**	69.5±3.8 (+4.7)
wine	24.0±18.5 (-68.0)	66.0±12.0 (-26.0)	90.0±0.0 (-2.0)	**92.0±4.0 (+0.0)**

Backbone models and baselines. By default, we use NTL as the backbone model, which was identified as state-of-the-art in a recent independent comparison of 13 models [167]. Results with other backbone models are shown in Appendix A.4. We implement the proposed LOE methods, the "Blind" and "Refine" baselines with NTL. For the "Refine" baseline and our methods, we set the number of warm-up epochs as two for small datasets and as one for large datasets. In warm-up epochs, the model is trained blindly.

For NTL, we use MLPs for neural transformations and the encoder on all datasets. We set the number of transformations as nine. The transformations are either parametrized with the network directly or as a residual connection of the network and the original sample. We search for the best-performed transformation parameterization and other hyperparameters based on the performance of the model trained on clean data. Further details are provided in Appendix B.3.

Empirical results. We report F1-scores for 30 tabular datasets in Table 4.3. The results are reported as the mean and standard derivation of five runs with different model initializations and random training set split. We set the contamination ratio $\alpha = \alpha_0 = 0.1$ for all datasets. LOE outperforms the "Blind" and "Refine" baselines consistently. Remarkably, LOE trained on contaminated data can achieve better results than on clean data on some datasets (see Table 4.3), suggesting that the latent anomalies provide a positive learning signal. This effect can be seen when increasing the contamination ratio on the Arrhythmia and Thyroid datasets (Figure 4.2 (c) and (d)). Hendrycks et al. [31] noticed a similar phenomenon when adding *labeled* outliers; these known anomalies help the model learn better region boundaries for normal data. Our results suggest that even *unlabelled* anomalies, when properly inferred, can improve the performance of an anomaly detector. Overall, we conclude that LOE significantly improves the performance of anomaly detection methods on contaminated tabular datasets. However, we also observed a large variance in the results on some datasets. For example, on Optidigits, LOE achieves either perfect detection or flipped detection in different runs. A flipped detection at the beginning of the training could mislead LOE.

4.1.2.4 Experiments on Video Data

In addition to image and tabular data, we also evaluate our methods on a video frame anomaly detection benchmark also studied in Pang et al. [168]. The goal is to identify video frames that contain unusual objects or abnormal events. Experiments show that our methods achieve state-of-the-art performance on this benchmark.

Video dataset. We study UCSD Peds1[3], a popular benchmark for video anomaly detection. It contains surveillance videos of a pedestrian walkway. Non-pedestrian and unusual behavior is labeled as abnormal. The data set contains 34 training video clips and 36 testing video clips, where all frames in the training set are normal and about half of the testing frames are abnormal. We follow the data preprocessing protocol of Pang et al. [168] for dividing the data into training and test sets. To realize different contamination ratios, we randomly remove some abnormal frames from the training set, but the test set is fixed.

[3]`http://www.svcl.ucsd.edu/projects/anomaly/dataset.htm`

Table 4.4: AUC (%) for video frame anomaly detection on UCSD Peds1 with various contamination ratios, including 10%, 20%, and 30%*. LOE$_S$ achieves the best results on ratios 10% and 20% and performs competitively with the state-of-the-art method Pang et al. [168].

Method	Contamination Ratio		
	10%	20%	30%*
Sugiyama and Borgwardt [169]	55.0	56.0	56.3
Pang et al. [168]	68.0	70.0	**71.7**
Blind	85.2±1.0	76.0±2.7	66.6±2.6
Refine	82.7±1.5	74.9±2.4	69.3±0.7
LOE$_H$ (ours)	82.3±1.6	59.6±3.8	56.8±9.5
LOE$_S$ (ours)	**86.8±1.2**	**79.2±1.3**	**71.5±2.4**

*Default setup in Pang et al. [168], $\alpha_0 \approx 30\%$.

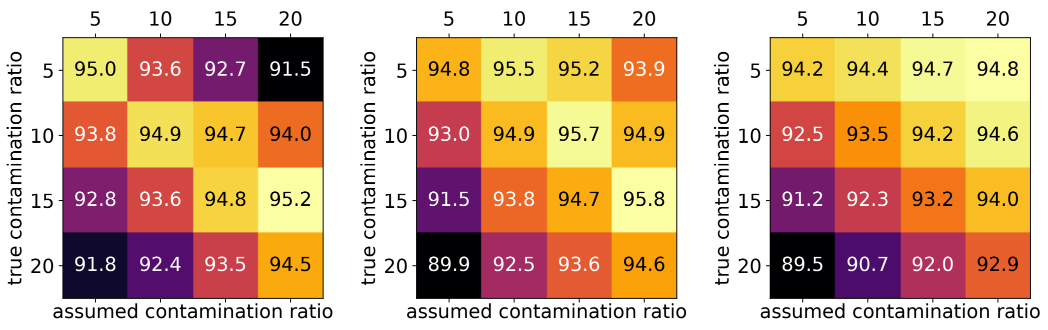
Backbone models and baselines. In addition to the "Blind" and "Refine" baselines, we compare to Sugiyama and Borgwardt [169], a distance-based method, and Pang et al. [168], an ordinal regression-based state-of-the-art method for video frame anomaly detection.We implement the proposed LOE methods, the "Blind", and the "Refine" baselines with NTL as the backbone model. We use a pre-trained ResNet50 on ImageNet as a feature extractor, whose output is then sent into an NTL. The feature extractor and NTL are jointly optimized during training. Further details are provided in Appendix B.3.

Empirical results. We report the results in Table 4.4. Our soft LOE method achieves the best performance across different contamination ratios. Our method outperforms Pang et al. [168] by 18.8% and 9.2% AUC on the contamination ratios of 10% and 20%, respectively. LOE$_S$ is also robust to the unusually large contamination ratio and performs competitively with the state-of-the-art method Pang et al. [168] when the ratio $\alpha_0 = 30\%$. LOE$_S$ outperforms the "Blind" and "Refine" baselines significantly on various contamination ratios from 10% to 30%.

4.1.2.5 Sensitivity Study

The hyperparameter α characterizes the assumed fraction of anomalies in our training data. Here, we evaluate its robustness under different ground truth contamination ratios. We run LOE$_H$, LOE$_S$, and the "Refine" baseline with NTL on CIFAR-10 with varying true anomaly ratios α_0 and different hyperparameters α. We present the results in a matrix accommodating the two variables. The diagonal values report the results when correctly setting the contamination ratio.

LOE$_H$ (Figure 4.3 (a)) shows considerable robustness: the method suffers at



(a) LOE_H (b) LOE_S (c) Refine

Figure 4.3: A sensitivity study of the robustness of LOE_H, LOE_S, and "Refine" to the misspecified contamination ratio. We evaluate them on CIFAR10 in terms of AUC (%). LOE_H and LOE_S yield robust results and outperform the "Refine" baseline in most cases.

most 1.4% performance degradation when the hyperparameter α is off by 5%, and is always better than "Blind". It always outperforms "Refine" (Figure 4.3 (c)) when erroneously setting a smaller α than the true ratio α_0. LOE_S (Figure 4.3 (b)) also shows robustness, especially when erroneously setting a larger α than the true ratio α_0. The method is always better than "Refine" (Figure 4.3 (c)) when the hyperparameter α is off by up to 15%, and always outperforms "Blind".

4.2 Active Latent Outlier Exposure

In some cases, expert feedback is available in learning the anomaly detector. For example, in a medical setting, one may ask a medical doctor to confirm whether a given image shows normal or abnormal cellular tissue. Other application areas include detecting network intrusions or machine failures. As expert feedback is typically expensive, it is essential to find effective strategies for querying informative data points. To this end, we study the active anomaly detection problem.

Active anomaly detection has been studied for decades. A variety of querying strategies have been proposed to specific shallow anomaly detectors such as OCSVM and KDE [170, 171, 171, 172, 173, 174, 175, 176]. However, shallow methods are known to be problematic for high-dimensional data [164]. Recently, deep active anomaly detection has received a lot of attention. Pimentel et al. [177] propose to query samples with the top anomaly scores for autoencoder-based methods, while Ning et al. [178] improve the querying with a diversity consideration. Tang et al. [179] use an ensemble of deep anomaly detectors and query the most likely abnormal samples for each detector separately. Russo et al. [180] query samples where the model is uncertain about the predictions. Pang et al. [181] and Zha et al. [182] propose querying strategies based on reinforcement learning, which requires labeled datasets for learning a Markov decision process.

These works primarily involved domain-specific applications and/or subpar architectures, making it hard to disentangle modeling choices from querying strategies. In this section, we aim to identify the different factors that lead to good performance in deep active anomaly detection. We find that diversified sampling strategies can dramatically improve popular querying strategies, such as querying data based on their predicted anomaly score or around the decision boundaries. Based on these findings, we propose ALOE: a new active learning strategy compatible with many self-supervised losses. ALOE queries data diversely in feature space based on the seeding algorithm of K-means++ and draws information from both queried and unqueried data based on two equally weighted losses. By just querying a few diverse samples, ALOE can efficiently estimate the sole hyperparameter in LOE, the assumed anomaly rate, based on an importance sampling estimate.

4.2.1 Algorithm and Methodology of ALOE

Active learning assumes that the training data contains unlabeled anomalies. However, querying the whole dataset (i.e., obtaining labels for all data) is usually infeasible due to the sheer amount of data and the high cost associated with getting expert feedback. Therefore, we can assume that we will only obtain labels for a small fraction of the data. Once some queries are labeled, the performance of an anomaly detector can be improved via the additional annotations. In active learning, it is much more common to seek a semi-supervised approach that accounts both for the labeled samples and the remaining unlabeled dataset. ALOE falls into this category. We first present our proposed loss and then discuss our proposed querying strategy.

4.2.1.1 Loss Function

We consider a joint distribution $p(\boldsymbol{x}, y)$ over data $\boldsymbol{x}$ and binary anomaly labels y, where $y = 0$ represents normal and $y = 1$ represents abnormal data. Our goal is to learn an anomaly detection model that associates every input with an anomaly score $s(\boldsymbol{x})$ that we can use to predict y for any $\boldsymbol{x}$. After querying, a subset of the data will be labeled and can be used to train the anomaly detector in a supervised manner.

Supervised loss. In order to use the normal data to train the model, we assume we have available a self-supervised or unsupervised loss function $\ell_n(\boldsymbol{x}; \theta)$, where θ denotes the model parameters. This could be an autoencoder loss [19], an OCC loss [89], or a NTL objective (Equation (2.2)). By construction, this loss tries to learn an anomaly detector based on only normal data. Similar to Hendrycks et al. [31] and

LOE in Section 4.1.1, we also require a corresponding loss function for abnormal data. This loss $\ell_a(\boldsymbol{x};\theta)$ shares parameters with the normal loss and is typically chosen to have a complementary effect on the learned function. Denoting our queried indices by $\mathcal{Q}$, we obtain a corresponding supervised loss as

$$\mathcal{L}_{\mathcal{Q}}^{\theta} = \frac{1}{|\mathcal{Q}|} \sum_{j \in \mathcal{Q}} (y_j \ell_a(\boldsymbol{x}_j;\theta) + (1 - y_j)\ell_n(\boldsymbol{x}_j;\theta)). \tag{4.9}$$

Instead of training this loss only on queried data, we improve on this idea and consider how to use both labeled and unlabeled samples to provide training signals. LOE have shown that this loss function is also applicable when the labels are unobserved as expectation-maximization can be used to infer the latent anomaly labels in Section 4.1.1.2.

ALOE loss. We propose to combine the supervised loss in Equation (4.9) with the unsupervised loss in Equation (4.1), thereby drawing information from both the queried and unqueried parts of the data. As follows, we assume that our data indices $\mathcal{I} = \{1, \cdots, N\}$ partition into a set of unlabeled data $\mathcal{U}$ and a set of queried data $\mathcal{Q}$ such that $\mathcal{I} = \mathcal{U} \cup \mathcal{Q}$. For all queried data, we assume that ground truth labels y_i are available, while for unqueried data, the labels $\tilde{y}_i$ are unknown. Adding both losses together yields

$$\mathcal{L}_{\text{ALOE}}^{\theta}(\tilde{\boldsymbol{y}}) = \mathcal{L}_{\mathcal{Q}}^{\theta} + \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} (\tilde{y}_i \ell_a(\boldsymbol{x}_i;\theta) + (1 - \tilde{y}_i)\ell_n(\boldsymbol{x}_i;\theta)). \tag{4.10}$$

Optimizing this loss involves a block coordinate ascent scheme that alternates between inferring the anomaly labels of unlabeled samples and taking gradient steps to minimize Equation (4.10) with the inferred labels. In each iteration, given $\mathcal{Q}$, the pseudo labels $\tilde{y}_i$ for $i \in \mathcal{U}$ are obtained by optimizing

$$\min_{\tilde{\boldsymbol{y}} \in \{0,0.5\}^{|\mathcal{U}|}} \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \tilde{y}_i \ell_a(\boldsymbol{x}_i;\theta) + (1 - \tilde{y}_i)\ell_n(\boldsymbol{x}_i;\theta) \quad \text{s.t.} \quad \sum_{i \in \mathcal{U}} \tilde{y}_i = 0.5(\alpha N - \sum_{j \in \mathcal{Q}} y_j).$$

The constraint on the labels ensures that the inferred anomaly labels respect a certain contamination ratio α, which is an important hyperparameter of the approach. In practice, this constrained optimization problem is solved as in Section 4.1.1.3 by using the current anomaly detector to rank the unlabeled samples based on their anomaly scores and assign the top α-quantile of the associated labels $\tilde{y}_i$ to the value 0.5, and the remaining to the value 0. Assigning $\tilde{y}_i = 0.5$ to the inferred anomalies accounts for the uncertainty of whether the sample truly is an anomaly.

In theory, α could be treated as a hyperparameter [162, 163], but eliminating hyperparameters is important in anomaly detection. In many practical applications of anomaly detection, there is no labeled data that can be used for validation.

While LOE have to assume that the contamination ratio is given, active learning provides an opportunity to estimate α. In Section 4.2.1.2, we develop an importance sampling-based approach to estimate α from the labeled data.

Another noteworthy aspect of the ALOE loss is that it weighs the *averaged* losses equally to each other. While equal weighting cannot be justified from the first principles, we expect our queried samples to be informative, and yet we do not want either loss component to dominate the learning task (assuming that we always query only a small percentage of the data). This provides more weight to every queried data point than to an unqueried one, assuring these goals. Our equal weighting scheme is also practical because it avoids a hyperparameter.

Equation (4.10) is our ALOE loss. It draws information from both labeled and unlabeled samples. Next, we propose how to select good queries for ALOE.

4.2.1.2 Active Querying

We consider the following active learning setup. An anomaly detection model is trained on a dataset involving unidentified anomalies. The trained model then uses some mechanism to select K data points $\boldsymbol{x}_i$ for querying an anomaly label $y_i \in \{0, 1\}$. Based on the queried samples, the model is then refined. During testing, we use only the learned anomaly detection model, i.e., we do not carry out active learning during testing, eliminating the need for a human in the loop.

Several active learning strategies for anomaly detection have been proposed in the literature. As a simple baseline, data points can be queried at random [22]; however, this oftentimes results in very few queried true anomalies. Görnitz et al. [173] proposed to query at the boundary of the normal region, which is the learned hypersphere surface in the feature space. A more general approach is to query data in the α-quantile of an anomaly detector, but the problem is that one typically does not know which value of α defines the decision boundary.

Our proposed solution is to query data points diversely, covering a large area in the feature space. This diverse querying has two benefits. First, we do not query data with redundant feature information. Second, it turns out that anomalies are oftentimes associated with regions of a greater distance to the center of the data. Thus, diverse querying will typically over-sample true anomalies, compensating for the fact that anomalies are rare in the training data.

Combining the requirements of informativeness and large support, we propose to use the seeding algorithm of K-means++ as our active learning strategy. This algorithm selects diverse initialization points for the K-means clustering. Specifically, we query samples in the unlabeled dataset $\mathcal{U}$ that are dissimilar to the samples in the queried set $\mathcal{Q}$. The query probability is proportional to the distance to its nearest sample in the queried set.

Algorithm 3: Diverse query of ALOE

```
Input: Unlabeled training dataset $\mathcal{U}$, querying budget $|\mathcal{Q}|$
Compute the pairwise distance $d(\boldsymbol{x}, \boldsymbol{x}')$ of samples in $\mathcal{U}$
Set the initial sample in the query set $\mathcal{Q} = [\arg\min_{\boldsymbol{x}} \sum_{\boldsymbol{x}' \in \mathcal{U}} d(\boldsymbol{x}, \boldsymbol{x}')]$
for $i = 2, \ldots, |\mathcal{Q}|$ do
    Sample $\boldsymbol{x}_i$ from $\mathcal{U}$ with the probability $p_{\text{query}}(\boldsymbol{x}_i)$
    $\mathcal{U} \leftarrow \mathcal{U} \setminus \boldsymbol{x}_i$
    $\mathcal{Q} \leftarrow \boldsymbol{x}_i \cup \mathcal{Q}$
end
```

For a more meaningful notion of distance, we define the latter in an embedding space as $d(\boldsymbol{x}, \boldsymbol{x}') = \exp(\|f^\theta(\boldsymbol{x}) - f^\theta(\boldsymbol{x}')\|_2/\tau)$, where f^θ is a neural feature map, and τ is a temperature parameter controlling the strength of diversity. We stress that all loss functions considered in this thesis already have an associated feature map that we can use. The distance of a sample $\boldsymbol{x}_i$ to the query set $\mathcal{Q}$ is defined as $D(\boldsymbol{x}_i) = \min_{\boldsymbol{x}_j \in \mathcal{Q}} d(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Given the existing queried samples $\mathcal{Q}$ we draw a sample from the unlabeled dataset $\mathcal{U}$ with a categorical distribution

$$p_{\text{query}}(\boldsymbol{x}_i) = \frac{D(\boldsymbol{x}_i)}{\sum_{\boldsymbol{x}_j \in \mathcal{U}} D(\boldsymbol{x}_j)} \quad \forall i \in \mathcal{U}. \tag{4.11}$$

Algorithm 3 summarizes our querying strategy.

4.2.1.3 Contamination Ratio Estimation

To eliminate a critical hyperparameter in our approach, we estimate the *contamination ratio*, i.e., the fraction of anomalies in the data set. Under a few assumptions, we show that this parameter can be estimated despite our biased querying strategy.

We consider the empirical fraction of anomalies in the training data, defined as $\alpha_0 = \sum_{i=1}^N \mathbb{1}_a(\boldsymbol{x}_i)/N$, where $\mathbb{1}_a(\boldsymbol{x}) \equiv \mathbb{1}(y(\boldsymbol{x}) = 1)$ indicates whether or not the sample is abnormal. Specifically, we want to estimate this quantity only based on our queried samples $\mathcal{Q}$, avoiding additional queries. Our querying procedure results in a chain of indices $\mathcal{Q} = \{i_1, i_2, ..., i_{|\mathcal{Q}|}\}$, where each conditional distribution $i_k | i_{<k}$ is defined by Equation (4.11). Unfortunately, the samples contained in $\mathcal{Q}$ are non-uniformly distributed, but importance sampling comes to the rescue. Importance sampling requires uncorrelated samples. Our first assumption is therefore as follows:

Assumption 1. *The anomaly scores $\{s(\boldsymbol{x}_i) : i \in \mathcal{Q}\}$ of queries are approximately independently distributed. Thus, they can be treated as independent samples in importance sampling.*

This assumption is only approximately valid, but we can justify it based on a dimensionality argument. By construction, the chain of queried data points

$\{\boldsymbol{x}_i : i \in \mathcal{Q}\}$ are spatially anti-correlated. However, we argue that these correlations get mostly lost in the projection step of $\boldsymbol{x}_i \mapsto s(\boldsymbol{x}_i)$. Consider repulsively sampling points in a n-dimensional hypercube, where n is large. Projecting these samples onto any one-dimensional subspace will result in approximately decorrelated samples since the samples still have $n-1$ dimensions to avoid each other.

We can now collect two data sets of one-dimensional scores, $\mathcal{S}_p = \{s(\boldsymbol{x}_i) : i \in \mathcal{I}\}$ (coming from the full data) and $\mathcal{S}_q = \{s(\boldsymbol{x}_i) : i \in \mathcal{Q}\}$ (coming from the queries). Since we are in one dimension, we can easily interpolate these discrete distributions to continuous ones using kernel density estimators with Gaussian kernels. Let $d_p(s)$ and $d_q(s)$ denote the resulting two densities. To set the bandwidth, we apply the average width as if the samples were evenly spaced in the range, i.e., $(\max_i(s_i) - \min_i(s_i))/|\mathcal{Q}|$.

To correct the sampling bias, we need to estimate the importance ratio between our proposal distribution $q(\boldsymbol{x}_i)$ (coming from the query) and the data distribution $p(\boldsymbol{x}_i)$. For tractability, we will do this in the space of anomaly scores. To this end, we make the following additional assumption.

Assumption 2. *The importance weights $p(\boldsymbol{x}_i)/q(\boldsymbol{x}_i)$ in the space of samples can be approximated by the importance weights in the anomaly score space, namely, $p(\boldsymbol{x}_i)/q(\boldsymbol{x}_i) \approx d_p(s(\boldsymbol{x}_i))/d_q(s(\boldsymbol{x}_i))$.*

We justify this assumption as follows. The learned anomaly score $s(\boldsymbol{x})$ of any deep anomaly detector is approximately a function of the data distribution's level sets. Thus, the anomaly score is (approximately) a sufficient statistic of the data distribution. We assume that the same holds true for the distribution of queried samples, which cannot differ too wildly from the data distribution.

Given both assumptions, we can estimate the contamination ratio based on our query set as

$$\alpha = \frac{1}{|\mathcal{Q}|} \sum_{i=1}^{|\mathcal{Q}|} \frac{p(\boldsymbol{x}_i)}{q(\boldsymbol{x}_i)} \mathbb{1}_a(\boldsymbol{x}_i) \approx \frac{1}{|\mathcal{Q}|} \sum_{i=1}^{|\mathcal{Q}|} \frac{d_p(s(\boldsymbol{x}_i))}{d_q(s(\boldsymbol{x}_i))} \mathbb{1}_a(\boldsymbol{x}_i). \tag{4.12}$$

Experiments show the bandwidth setup mentioned above works well in practice.

4.2.2 Experiments of ALOE

We study active anomaly detection on standard image benchmarks, medical images, and tabular data. Our extensive empirical study establishes how our proposed method compares to eight recent active anomaly detection methods we implemented as baselines. We find that diverse querying with K-means++ improves the detection accuracy of a broad range of self-supervised anomaly detection methods and that our method ALOE consistently outperforms all other methods by a large margin.

We first describe the baselines and their implementations. Afterward, we present the experiments on images and tabular data. Finally, we provide an analysis of the contamination ratio estimation. We also provide an ablation study showing the benefit of each ingredient in Appendix A.5.

Baselines. Most existing baselines apply their proposed querying and training strategies to shallow anomaly detection methods or sub-optimal deep models (e.g., autoencoders [19]). In recent years, these approaches have consistently been outperformed by self-supervised anomaly detection methods [39]. For a fair comparison of querying strategies, we endow all baselines with the same state-of-the-art self-supervised backbone models also used in our method. By default, we use NTL as the backbone model. Results with other backbone models are provided in Appendix A.5.

The baselines are summarized in Table 4.5 and detailed in Appendix B.4. They differ in the treatment of labeled and unlabeled samples (training losses in col. 4 & col. 5): the unlabeled data is either ignored or modeled with a one-class objective. Most baselines incorporate the labeled data by a supervised loss (Equation (4.9)). As an exception, Ning et al. [178] use data refinement [19, 157] to remove all labeled anomalies and then train a weighted one-class objective on the remaining data. All baselines weigh the unsupervised and supervised losses equally. The baselines also differ in their querying strategies (col. 3), summarized below:

- **margin query** selects samples deterministically that are the closest to the boundary of the normality region. We provide the method with the true contamination ratio to help it choose an ideal boundary.
- **margin diverse query** combines margin query with neighborhood-based diversification. It tends to select samples at different positions of the normality boundary. The final criterion is an equally weighted combination of the two aspects.
- **random query** draws samples uniformly among the training set.
- **most positive query** selects the top-ranked samples ordered by anomaly scores.
- **positive random query** samples uniformly among the top 50% data ranked by their anomaly scores.
- **positive diverse query** combines querying according to anomaly scores with distance-based diversification. The selection criterion is an equally weighted combination of the two aspects: anomaly score and the minimum Euclidean distance to all previously queried samples.

Implementation details. In all experiments, we use NTL as the backbone model for all methods. On image datasets, we apply NTL to the visual features extracted with a frozen ResNet152 pre-trained on ImageNet. On tabular datasets, we directly apply NTL to the data. We apply the same number of transformations, network

Table 4.5: A summary of all compared experimental methods' query strategies and training strategies irrespective of their backbone models.

Name	Reference	Querying	Loss (labeled)	Loss (unlabeled)
Mar	Görnitz et al. [173]	margin	superv. (Equation (4.9))	one class
Hybr1	Görnitz et al. [173]	margin diverse	superv. (Equation (4.9))	one class
Pos1	Pimentel et al. [177]	most positive	superv. (Equation (4.9))	none
Pos2	Barnabé-Lortie et al. [183]	most positive	superv. (Equation (4.9))	one class
Rand1	Ruff et al. [22]	random	superv. (Equation (4.9))	one class
Rand2	Trittenbach et al. [184]	positive random	superv. (Equation (4.9))	one class
Hybr2	Das et al. [176]	positive diverse	superv. (Equation (4.9))	none
Hybr3	Ning et al. [178]	positive diverse	refinement	weighted one class
ALOE	[ours]	K-means++	ALOE loss (Equation (4.10))	

components, and anomaly loss function ℓ_a as in Section 4.1.2. NTL is trained for one epoch, after which $|\mathcal{Q}|$ queries are labeled and collected at once. The contamination ratio α in ALOE is estimated using the labeled samples immediately after the querying step and then fixed for the remaining training process. We set $\tilde{y}_i = 0$ for samples that are inferred to be normal and $\tilde{y}_i = 0.5$ for inferred anomalies. More implementation details are provided in Appendix B.4.

4.2.2.1 Experiments on Image Data

In this section, we study ALOE on standard image benchmarks to establish how it compares to eight well-known active anomaly detection baselines. Active learning plays an important role in medical domains where expert labeling is expensive. Hence, we also study nine medical datasets from Yang et al. [185]. We describe the datasets, the evaluation protocol, and finally the results of our study.

Image benchmark datasets. First, we experiment with two popular image benchmark datasets: CIFAR10 and FashionMNIST. These have been used in virtually all deep anomaly detection papers published at top-tier venues [37, 38, 39, 41, 88, 89].

Medical image datasets. Since medical imaging is a very important practical application of active anomaly detection, we also study ALOE on medical images. The datasets we consider cover different data modalities (e.g., X-ray, CT, electron microscope), and their characteristic image features can be very different from natural images. Our empirical study includes all 2D image datasets presented in Yang et al. [185] that have more than 500 samples in each class, including BloodMNIST, OrganAMNIST, OrganCMNIST, OrganSMNIST, OCTMNIST, PathMNIST, PneumoniaMNIST, and TissueMNIST. We also include DermaMNIST but are restricted

Table 4.6: AUC (%) with standard deviation for anomaly detection on 11 image datasets when the query budget $|\mathcal{Q}|= 20$. ALOE outperforms all baselines by a large margin by querying diverse and informative samples.

	Mar	Hybr1	Pos1	Pos2	Rand1	Rand2	Hybr2	Hybr3	ALOE
CIFAR10	92.4±0.7	92.0±0.7	93.4±0.5	92.1±0.7	89.2±3.2	91.4±1.0	85.1±2.2	71.8±7.4	**96.3±0.3**
FMNIST	93.1±0.4	92.6±0.4	92.2±0.6	89.3±1.0	84.0±3.8	90.6±1.1	88.7±1.4	82.6±4.3	**94.8±0.6**
Blood	68.6±1.8	69.1±1.3	69.6±1.8	72.2±4.9	70.6±1.6	69.2±1.7	72.2±2.7	58.3±5.2	**80.5±0.5**
OrganA	86.4±1.3	87.4±0.7	81.7±2.9	81.8±2.1	82.9±0.6	86.5±0.7	88.6±1.5	68.8±3.1	**90.7±0.7**
OrganC	86.5±0.9	87.0±0.7	84.6±1.9	79.6±2.0	85.5±0.9	86.4±0.8	84.8±1.2	68.9±3.0	**89.7±0.7**
OrganS	83.5±1.1	84.1±0.4	83.2±1.3	78.6±1.0	82.2±1.4	83.8±0.4	82.3±0.7	66.9±4.3	**87.4±0.8**
OCT	64.4±3.7	63.3±1.8	63.8±4.4	63.0±4.0	59.7±1.9	62.1±4.3	63.0±7.6	56.2±4.5	**68.5±3.4**
Path	82.7±2.4	86.0±1.1	77.5±2.0	80.2±3.5	83.2±1.6	83.9±2.9	86.1±2.0	75.1±4.2	**88.1±1.1**
Pneu.	72.1±7.0	75.1±5.3	75.5±8.8	83.6±6.1	68.1±5.9	76.0±8.0	88.4±3.3	63.4±17.7	**91.2±1.4**
Tissue	60.2±1.5	61.3±1.7	65.8±1.7	63.5±2.0	59.9±1.7	59.5±1.3	62.1±1.7	50.8±1.6	**66.4±1.4**
Derma	62.6±3.8	63.1±4.7	66.6±2.3	66.4±4.3	64.5±4.8	68.3±2.1	57.2±13.3	48.0±13.6	**73.5±2.5**
Average	77.5	78.3	77.3	77.6	75.4	78.0	78.0	64.6	**84.3**

to the classes which have more than 500 training samples. The specific data statistics and the collection procedure are listed in Yang et al. [185].

Evaluation protocol. We follow the community standard known as the "one-vs.-rest" protocol to turn these classification datasets into a test-bed for anomaly detection [37, 38, 39, 41, 89]. While respecting the original train and test split of these datasets, the protocol iterates over the classes and treats each class in turn as normal. Random samples from the other classes are used to contaminate the data. The training set is then a mixture of unlabeled normal and abnormal samples with a contamination ratio of 10% [40]. This protocol can simulate a "human expert" to provide labels for the queried samples because the datasets provide ground-truth class labels. The models are trained on the training data and evaluated on the test data. The reported results for each dataset are averaged over the number of experiments (i.e., classes) and over five independent runs. Results are reported in terms of average AUC (%) with standard deviation.

Empirical results. We report the evaluation results of our method (ALOE) and the eight baselines on all 11 image datasets in Table 4.6. All methods have a query budget of 20 samples. On all datasets, our proposed method ALOE achieves the best performance and significantly outperforms all baselines by six percentage points on average, reaching state-of-the-art results in active anomaly detection on images.

In addition, we also study detection performance as the query budget increases from 20 to 160. The results are plotted in Figure 4.4. This systematic experiment demonstrates ALOE's superior performance is consistent given various active learning budgets. Even with a small budget of 20 samples, ALOE (by querying diverse and

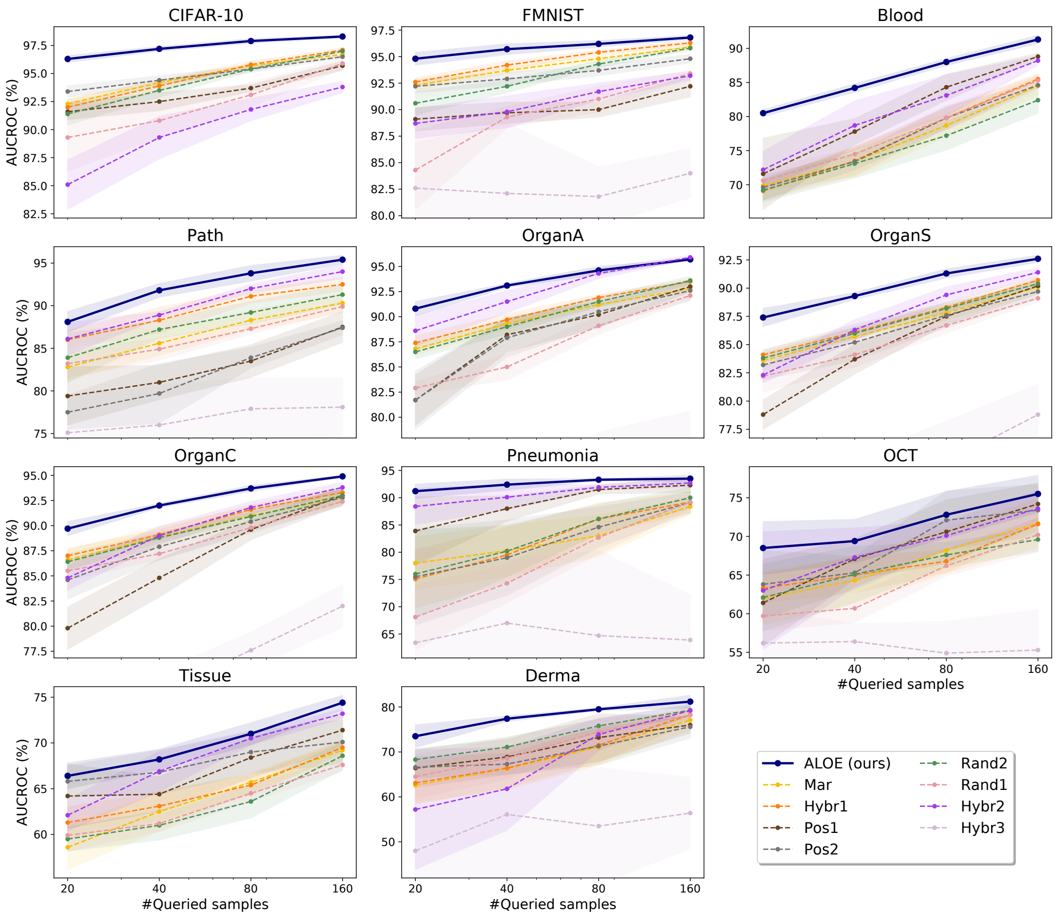


Figure 4.4: Results in terms of AUCs (%) for anomaly detection on 11 image datasets with the query budgets varying from 20 to 160. ALOE performs the best among the compared methods on all query budgets.

informative samples) makes better usage of the labeling information than the other baselines and thus leads to better performance by a large margin. The performance of almost all methods increases as more samples are allowed to be queried. Even for $|\mathcal{Q}|= 160$ queries where the benefit from adding more queries starts to saturate, ALOE still outperforms the baselines. We also provide results with other backbone models in Appendix A.5.

4.2.2.2 Experiments on Tabular Data

Many practical use cases of anomaly detection, for example, in healthcare or cyber security, are concerned with tabular data. For this reason, we study ALOE on a number of tabular datasets from various domains. We find that it outperforms

Table 4.7: F1-score (%) with standard error for anomaly detection on tabular datasets when the query budget $|\mathcal{Q}|= 10$. ALOE performs the best on 3 of 4 datasets. We set the contamination ratio as 10%.

	Mar	Hybr1	Pos1	Pos2	Rand1	Rand2	Hybr2	Hybr3	ALOE
BreastW	81.6±0.7	83.3±2.0	58.6±7.7	81.3±0.8	87.1±1.0	82.9±1.1	55.0±6.0	79.6±4.9	**93.9±0.5**
Iono.	91.9±0.3	**92.3±0.5**	56.1±6.2	91.1±0.8	91.1±0.3	91.9±0.6	64.0±4.6	88.2±0.9	91.8±1.1
Pima	50.1±1.3	49.2±1.9	48.5±0.4	52.4±0.8	53.6±1.1	51.9±2.0	53.8±4.0	48.4±0.7	**55.5±1.2**
Satellite	64.2±1.2	66.2±1.7	57.0±3.0	56.7±3.2	67.7±1.2	66.6±0.8	48.6±6.9	56.9±7.0	**71.1±1.7**
Average	72.0	72.8	55.1	70.4	74.9	73.3	55.4	68.3	**78.1**

existing baselines, even with as few as ten queries.

Tabular datasets. Our study includes four multi-dimensional tabular datasets from the ODDS repository [4], which have an outlier ratio of at least 30%. This is necessary to ensure that there are enough anomalies available to remove from the test set and add to the clean training set to achieve a contamination ratio of 10%. The datasets are BreastW, Ionosphere, Pima, and Satellite. To form the training and test set for tabular data, we first split the data into normal and abnormal categories. We randomly sub-sample half the normal data as the training data and treat the other half as the test data. To contaminate training data, we randomly sub-sample the abnormal data into the training set to reach the desired 10% contamination ratio; the remaining abnormal data goes into the test set.

As in the image experiments, there is one round of querying after one-epoch training, in which 10 samples are labeled. We evaluate the model performance of all baselines in terms of F1-score (%). For each dataset, we report the averaged results with standard errors over five runs with random train-test splits and random initialization.

Empirical results. We report the results for our method in comparison to the eight baselines (described in Table 4.5) in Table 4.7. Our proposed method ALOE performs best on three of four datasets and outperforms all baselines by at least 3.2 percentage points on average. Diverse querying best utilizes the query budget to label the diverse and informative data points, yielding a consistent improvement over existing baselines in active anomaly detection on tabular data.

4.2.2.3 Analysis on Ratio Estimation

We analyze the accuracy of the contamination ratio estimation (Equation (4.12)). The relative errors of the estimated contamination ratio to the true ratio when the

[4] `http://odds.cs.stonybrook.edu/`

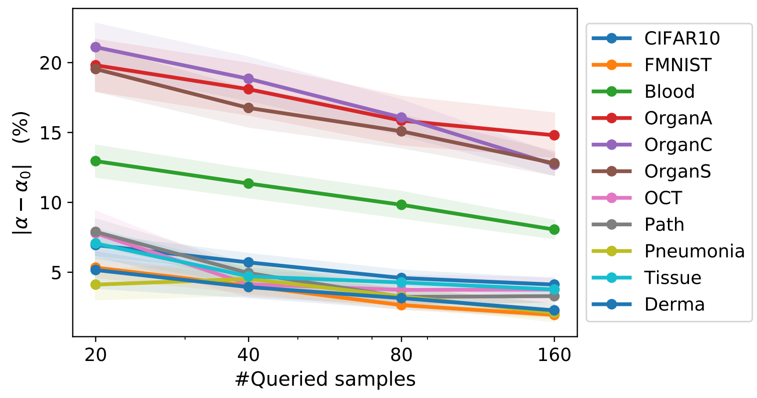


Figure 4.5: Relative error (%) of the estimated contamination ratio $\tilde{\alpha}$ to the true ratio α_0 on all image datasets. The true contamination ratio is 10%. The estimation error is less than 10% on 7 of 11 datasets, even when querying 20 samples.

Table 4.8: AUC(%) with standard error of ALOE with estimated ratio and ALOE with true ratio when the query budget $|\mathcal{Q}| = 20, 40, 80, 160$. The results are averaged over 11 image datasets. ALOE with either true ratio or estimated ratio performs similarly on all query budgets.

K	20	40	80	160
ALOE	84.3±3.0	86.4±2.9	88.4±2.7	89.9±2.5
ALOE (true ratio)	85.2±3.0	87.0±3.0	88.7±2.7	90.1±2.5

query budget $|\mathcal{Q}| = 20, 40, 80, 160$ are reported in Figure 4.5. The estimation error is less than 10% on 7 of 11 datasets, even when the query budget is just 20. When the query budget increases, the estimation error goes smaller on all datasets as expected. Since LOE (the training strategy on the unqueried samples) is robust against the misspecified ratios as shown in Section 4.1.2, we found that the estimation error does not cause a clear degradation in all experiments.

We compare ALOE to the counterpart with the true anomaly ratio to see how the estimated ratio affects the detection performance. The results are reported in terms of AUC (%) with standard error averaged over 11 image datasets in Table 4.8. The performance degradation caused by a misspecified ratio is maximal 0.9% for a query budget of 20 and decreases to 0.2% when querying 160 samples. It shows that ALOE with either true ratio or estimated ratio performs similarly with all query budgets. Using an estimated ratio only leads to negligible performance degradation. Therefore, the estimated ratio can be applied safely. This is very important in practice since the true anomaly ratio is unknown in many applications.

4.3 Summary and Discussion

We first propose LOE: a domain-independent approach for training deep anomaly detectors on a dataset contaminated by unidentified anomalies in an unsupervised manner. During training, LOE infers anomalous data in the training set while updating model parameters by solving a mixed continuous-discrete optimization problem; iteratively updating the model and its predicted anomalies. Similar to outlier exposure [31], LOE extracts learning signals from both normal and abnormal samples by considering a combination of two losses for both normal and (assumed) abnormal data, respectively. Our approach can be applied to a variety of anomaly detection benchmarks and loss functions. As demonstrated in our comprehensive empirical study, LOE yields significant performance improvements on all image, tabular, and video data.

We then propose ALOE: an active learning approach for querying anomaly labels and training deep anomaly detection models if expert feedback is available. ALOE relies on a diversified querying strategy based on the seeding algorithm of K-means++ and a combination of two losses for queried and unqueried samples. Based on a simple heuristic for weighting the losses relative to each other and by estimating the unknown contamination rate from queried samples, we were able to make ALOE free of the most important hyperparameter in LOE. We showed on a variety of datasets from different domains that the approach results in a new state-of-the-art in active anomaly detection.

LOE has an important hyperparameter, the assumed contamination ratio. The estimation of the contamination ratio relies on prior knowledge or the expert's experience in the unsupervised scenario. ALOE can eliminate this sole hyperparameter efficiently based on an importance-weighted estimate when expert feedback is available to label a few informative samples. However, the success of the estimation relies on several heuristics that cannot be proven rigorously. These heuristics include our relative weighting of the averaged losses for the queried and unqueried data, as well as our estimation procedure for the importance weights. Nevertheless, the fact that these assumptions were validated on a variety of data makes us optimistic that these heuristics will generalize to other domains. Furthermore, estimating the contamination ratio can be noisy when the query set is small. But note that LOE is shown to be robust even under the misspecification of the contamination ratio. Finally, the diversified sampling strategy becomes expensive when the dataset is large. This shortcoming can be mitigated by random data thinning before querying.

Parts of this chapter are mainly based on:

[186] **Chen Qiu***, Aodong Li*, Marius Kloft, Maja Rudolph, and Stephan Mandt. Latent Outlier Exposure for Anomaly Detection with Contaminated Data. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of Proceedings of Machine Learning Research, pages 18153–18167. PMLR, 17–23 Jul 2022.

Aodong Li*, **Chen Qiu***, Padhraic Smyth, Marius Kloft, Stephan Mandt, and Maja Rudolph. Diverse Querying Improves Deep Active Anomaly Detection. *Preprint (under review)*, 2022.

5 Conclusion and Outlook

We conclude this thesis with a summary of the results in Section 5.1 and an outline of the future research paths in Section 5.2.

5.1 Summary of Results

This thesis has contributed NTL, a self-supervised anomaly detection approach with learnable transformations. NTL learns a variety of semantically meaningful and diverse transformations from data and uses the learned transformations for detecting anomalies. Thanks to the flexibility of the learnable transformations, NTL is applicable to various data types, including time series, images, tabular data, and text. The key ingredient is a novel training objective that manages the trade-off between the semantics and the diversity of the learnable transformations. We have theoretically demonstrated that the proposed loss function learns semantically meaningful and diverse transformations and is better suited than existing self-supervised losses. We have also critically discussed the theoretical limitations of NTL in learning transformations and the score function. Implied by the constructed failure cases, we have provided practical recommendations on the architecture design and implementation. In the experimental evaluation, we found that NTL improves over many strong baselines in anomaly detection on sequences, images, tabular data, text, and also anomaly localization on images.

NTL can also be a plug-in that complements other components in a system to enhance the anomaly detection performance. We have introduced extensions of NTL, which combine with representation learning or deep OCC. For detecting anomalies within time series, we have introduced LNT, which combines NTL and time series representation learning to learn local transformations and produce an anomaly score for each time step. We have proven that both NTL and representation learning complement each other to avoid trivial solutions not appropriate for anomaly detection within time series. In experiments, we found that LNT can detect hard anomalies in time series with complex dynamics. For graph-level anomaly detection, we have introduced OCGTL and MOCC. Both methods combine NTL and deep

OCC and become more flexible and more expressive. Our extensive experiments have revealed that OCGTL and MOCC have no hypersphere collapse and raise the bar in graph-level anomaly detection significantly.

Furthermore, we have studied a more practical scenario of anomaly detection, where the training set is contaminated by unnoticed anomalies. We have shown that naive training on contaminated data leads to performance degradation. Addressing this, we have introduced a novel training strategy, LOE, which infers unlabeled anomalies and uses them to provide positive training signals. In our extensive experiments, we found that LOE is applicable to various deep anomaly detection models and significantly improves over established baselines in anomaly detection on images, video, and tabular data. With ALOE, we have extended LOE to the active anomaly detection setting. ALOE queries diverse samples based on the seeding algorithm of K-means++ and learns the model parameters using both queried and unqueried data. One additional benefit is that ALOE eliminates this sole hyperparameter in LOE given a few queried samples. We showed on a variety of image and tabular datasets that ALOE results in a new state-of-the-art in active anomaly detection.

In conclusion, this thesis has demonstrated through various experiments and analyses that learning transformations and detecting anomalies with NTL improves over the state-of-the-art on general data types and various applications. Moreover, for the more practical scenario of anomaly detection with contaminated data, this thesis has demonstrated through extensive experiments that NTL becomes robust to data contamination with the proposed training strategy LOE or the proposed active learning approach ALOE. In the remainder of this thesis, we turn to what lies ahead and identify specific paths for future research.

5.2 Future Research Paths

We discuss the potential future research directions in the following.

5.2.1 Multimodal Anomaly Detection

The different data types describe the same object from various perspectives generally. Learning to exploit different data types together can lead to a more powerful detection system. For example, driver fatigue detection relying on a fusion of EEG signals, gyroscope, and images is more accurate than the detection solely based on facial expressions [187]. Overall, multimodal anomaly detection is interesting yet under-explored. So one of the next steps of building a powerful anomaly detector is to fuse the information from different data sources for anomaly detection.

Learning a multimodal anomaly detector requires solving three tasks: extracting features from various data types, feature fusion, and anomaly detection. Many existing representation learning methods have shown great success in learning good features in an unsupervised manner, e.g., BERT [55] on language data, CPC [62] on time series, and masked autoencoders [188] on visual data. They provide a solid foundation for solving feature extraction. For unsupervised feature fusion, simple operations, such as summation or concatenation, might not be the optimal solution. The attention mechanism is shown to be a better operation of fusing the features in many applications [189, 190, 191, 192]. To sum up, a simple two-stage multimodal anomaly detector consists of pre-trained feature extractors for each data source, an attention-based fusion operation, and a simple downstream anomaly detector (e.g., OCSVM, KNN). Also, we can design an end-to-end multimodal anomaly detector that solves feature extraction, feature fusion, and anomaly detection jointly. Notice that these data sources are different yet not isolated from each other. Learning their potential commonalities and discrepancies is vital for both feature extraction and feature fusion. To do so, we might borrow ideas from self-supervised learning by treating each data source as a view. The model learns and fuses the features from different data sources by contrasting them or predicting them in the auxiliary task. The anomalies can then be scored by the model performance in the auxiliary task.

5.2.2 Meta-Learning Meets Anomaly Detection

In real-world scenarios, the distribution of data often changes according to the environment. The distribution shift may change the scope of normal samples or the definition of anomalies. For example, if the model is tested on the samples collected by sensors (e.g., cameras) different from the sensors used in training, the model performance might drop significantly. Generalizing to unseen data distribution often challenges the data-driven anomaly detection approaches. The classical learning setup expects to learn a single model covering all environments or learn environment-specific models. The former neglects the discrepancy of environments and leads to sub-optimal results in general, while the latter requires a large amount of data for each environment which is not practical in real-world applications.

Meta-learning suggests learning a metal model that captures the commonalities of various environments and adapts to the test environment efficiently. Lu et al. [193], Wu et al. [194] propose to learn an adaptive anomaly detector with MAML algorithm [195] for image anomaly detection. During the test, the model is rapidly fine-tuned by a few samples from the test distribution. Meta-training requires a training set involving samples from multiple tasks or distributions. However, it is impractical for one-class classification learning, where the samples are commonly assumed to be from the same normal distribution. This conflict makes the employment of meta-training in anomaly detection not straightforward in real-world applications, especially when the training data is scarce. Addressing this, we might use data augmentations to simulate multiple data distributions in meta-training. By learning to adapt across the augmented data distributions, we expect the meta-model to capture the common features of the normal samples and be able to adapt to new distributions efficiently.

5.2.3 Continual Anomaly Detection

Most previous works assume that the training data covers all normal data patterns and relies on offline training. However, we might receive training samples continually in practice, and the new incoming normal samples follow a data distribution different from the previously observed one. For example, a manufacturing chain collects data from a new sensor system every day. The data from the new sensor system is still normal but might be from a different distribution. Apart from this, the definition of anomalies might change in the new stage. For instance, a surveillance system records video frames under new weather when the season changes. Wearing heavy clothes is normal in the winter but is less normal in the summer. This raises a research question: can we design a continual learning procedure that updates deep anomaly detectors sorely on new incoming samples without forgetting the learned knowledge? By studying this question, we aim to build a closed-loop of data and anomaly detectors.

Wiewel and Yang [196] first study anomaly detection with replay-based continual learning, where a variational autoencoder is learned to generate samples for complementing the new incoming training samples. Doshi and Yilmaz [197, 198] propose memory-based continual anomaly detection methods for the surveillance video application. A meta-learning algorithm is proposed for continual anomaly detection by Frikha et al. [199], while a regularization-based continual anomaly detection method is introduced by Maschler et al. [200]. Although continual anomaly detection has been explored in several existing works, a standard benchmark is not established. The existing continual learning algorithms can be categorized to replay methods, regularization-based methods, and parameter isolation methods [201]. By building a benchmark with well-controlled experiment setups, we can first answer what the suitable continual learning algorithm for anomaly detection is. This can also pave the way for designing continual learning algorithms for anomaly detection specifically.

5.2.4 Representation Learning for Anomaly Detection

The quality of data representations influences the performance of machine learning algorithms heavily. A good representation reveals the useful semantic information of the data and makes the downstream tasks easier. Deep learning automates feature learning. The advanced representation learning methods [73, 74, 202] have improved the model performance on various downstream tasks. However, learning rich features is not straightforward due to the lack of objective signals in anomaly detection.

Self-supervised anomaly detection methods enhance representation learning and scoring by exploiting data augmentations. Sohn et al. [37] learn representations with contrastive learning and evaluate the learned representations for anomaly detection. Contrastive learning promotes the uniformity of the representations and therefore learns linearly separable representations [203]. The uniformity of the representations leads to performance improvement in multi-class classification. However, they found that uniformly distributed representations of normal samples are not optimal for anomaly detection or one-class classification since the anomalies cannot be easily isolated then.

Addressing this, Sohn et al. [37] learn compact representations of normal samples by contrasting them to rotated images. Similarly, Reiss and Hoshen [87] propose to learn compact representations by combining contrastive learning and OCC loss. Both methods try to improve the representations' quality of contrastive learning by alleviating the uniformity property of contrastive learning. Also, the success of contrastive learning relies on well-designed data augmentations and cannot be generalized to other data types easily. To alleviate the dependence of contrastive learning on data augmentation, Tamkin et al. [120], Shi et al. [204] coincide with the idea of learning transformation via adversarial training. In the broader context, an interesting question will be is contrastive learning the most suited algorithm for learning representations for anomaly detection on general data types.

Bert [55] learns effective text representations by learning to predict the masked tokens. He et al. [188] extend the self-prediction idea to images and learn visual representations by predicting the masked image patches. The self-prediction-based learning framework has shown success in both visual and language domains and on various downstream tasks except anomaly detection. Different than contrastive learning, the self-prediction-based learning framework does not enforce the uniformity of representations. So one remaining open question is how well the learned representations for anomaly detection on different data types work.

5.2.5 Explainable Anomaly Detection

Deep anomaly detection methods have demonstrated their ability in terms of achieving high accuracy on a variety of tasks. However, the use of these black-box models outside the machine learning community has been hampered by difficulties in interpreting them. The research on explainable anomaly detection has recently received much attention [205, 206, 207]. The core motivation here is to understand why a sample is labeled as an anomaly. Numerous algorithms have been proposed for anomaly segmentation of an image, which reveals the salient abnormal regions [88, 108, 109, 110, 111, 208, 209, 210, 211]. Beyond that, Du et al. [212] detect out-of-distribution objects in an image. For example, the appearance of pedestrians is normal on the sidewalk but is abnormal on the highway. The anomaly is caused by the abnormal co-occurrence of the object and the background scenario. Also, in medical diagnosis, it is crucial to understand what causes the diagnosis. Figuring out this causal relationship guarantees the reliability of the diagnosis and hints at the therapy targets [213]. These examples suggest the development of an anomaly detection method that can understand the underlying correlation or even causal relationship of the inputs. Moreover, building logical rules or symbolic formulas that describe the black-box deep anomaly detector can even achieve a higher level of explainability.

Appendix

A Supplementary Analysis and Results

A.1 Visual Analysis of NTL on Synthetic Data

We analyze NTL on a synthetic dataset. For the sake of visualization, we created a 2D dataset with two circles. The inner-circle (blue) is considered normal, and the outside circle (orange) is abnormal (see Figure A.1). The model is trained on samples drawn from the inner circle. The neural transformations $\mathcal{T}_k^\theta$ with $k \in \{1, \cdots, K\}$ are modeled by K feed-forward neural networks and the transformation $\mathcal{T}_0^\theta$ is an identity transformation. We set the number of learnable transformations $K = 4$.

In the first row of Figure A.1, we visualize the gradient vector fields of how $\mathcal{T}_{1:K}^\theta$ transform normal samples and anomalies. The length of the arrow indicates the transformation magnitude. In the second row of Figure A.1, we visualize the anomaly score contour lines, where the dark areas have low scores, and the light color corresponds to a high score. $s_k(x)$ denotes the score term led by the transformation $\mathcal{T}_k^\theta$ and is thus defined as

$$s_k(x) = -\log \frac{h(\mathcal{T}_k^\theta(x), \mathcal{T}_0^\theta(x))}{\sum_{l \in \{0,\ldots,K\} \setminus \{k\}} h(\mathcal{T}_k^\theta(x), \mathcal{T}_l^\theta(x))}. \tag{A.1}$$

Apart from the score terms $s_k(x)$, we also visualize the complete score function $s(x)$ involving all score terms $s_{1:K}(x)$. The transformations $\mathcal{T}_{1:4}^\theta$ learn salient features of normal samples and move normal samples to different directions. Since anomalies do not share the same features with normal samples, the transformations move anomalies to similar directions and lead to higher anomaly scores. From the sub-figures of $s_{1:4}(x)$, we can see each transformation learns to predict low anomaly scores on normal regions and high anomaly scores on different anomalous regions. The complete score function $s(x)$ predicts the contour line encompassing the normal regions successfully by taking all transformations and their associated score terms into account. In summary, each transformation learns different salient features of normal samples and helps the model detect anomalies from a different view. The

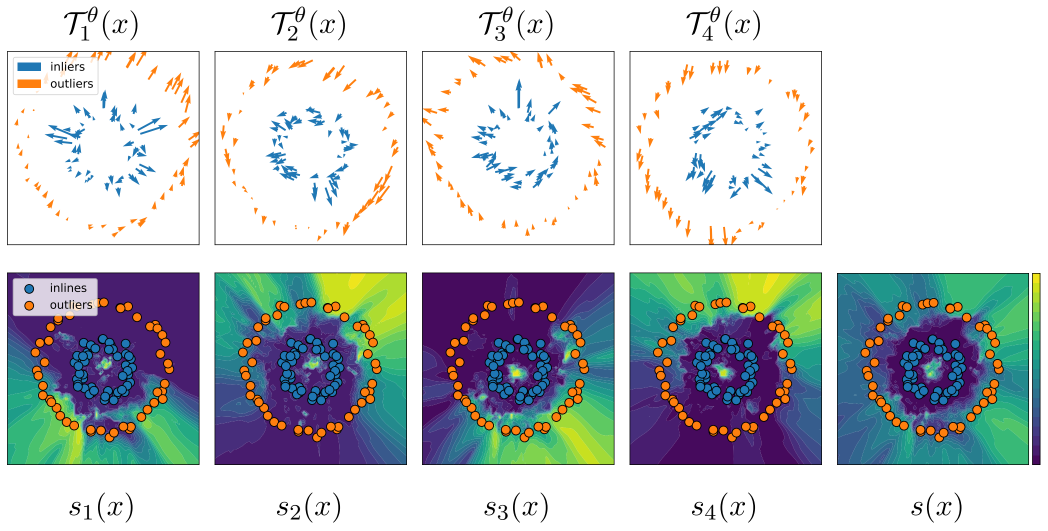


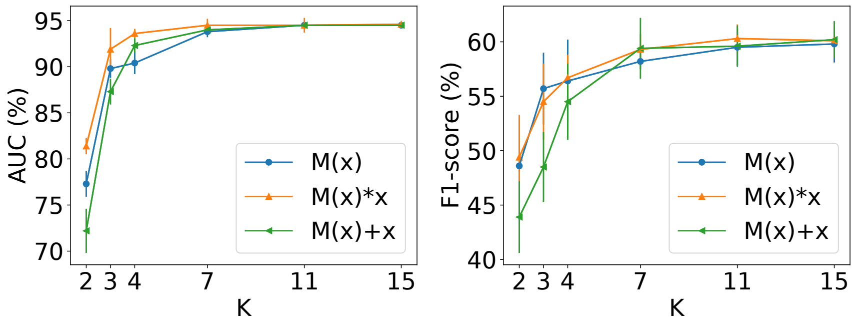
Figure A.1: Visualization of learned transformations and anomaly score contour lines on 2D synthetic data with two circles. The data points (blue) in the inner-circle are considered normal, and the points (orange) in the outside circle are abnormal. We visualize the gradient vector fields of learned transformations in the first row. In the second row, we visualize the score terms $s_k(x)$ led by each transformation $\mathcal{T}_k^\theta$ and the complete anomaly score $s(x)$.

anomaly score $s(x)$ earns a complete sense of normal and anomalous patterns by considering various transformations.

A.2 Ablation Study of NTL

We study the performance of NTL under various design choices for the learnable transformations, including their parametrizations, and their number K. We consider the following parametrizations: *forward* $\phi_k^\theta(\boldsymbol{x}) = M_k^\theta(\boldsymbol{x})$, *residual* $\phi_k^\theta(\boldsymbol{x}) = M_k^\theta(\boldsymbol{x}) + \boldsymbol{x}$, and *multiplicative* $\phi_k^\theta(\boldsymbol{x}) = M_k^\theta(\boldsymbol{x}) \odot \boldsymbol{x}$, which differ in how they combine the learnable masks $M_k^\theta(\cdot)$ with the data.

In Figure A.2 we show the anomaly detection accuracy achieved with each parametrization, as K varies from 2 to 15 on the time series data NATOPS and the tabular data Arrhythmia. For large enough K, NTL is robust to the different parametrizations, since DCL ensures the learned transformations satisfy the semantic requirement and the diversity requirement. The performance of NTL improves as the number k increases. When K is large enough the performance becomes stable. When $K \leq 4$, the performance has a larger variance, since the learned transformations are not guaranteed to be useful for anomaly detection without the guidance of



(a) AUC on NATOPS (b) F1-score on Arrhythmia

Figure A.2: Anomaly detection accuracy in terms of AUC of NTL on NATOPS and in terms of F1-score of NTL on Arrhythmia increases as the number of transformations K increases, but stabilizes when a certain threshold is reached ($K >\approx 10$). With enough transformations, NTL is robust to the transformation parametrization.

any labels. When K is large enough, the learned transformations contain with a high likelihood some transformations that are useful for anomaly detection. The transformation-based methods (including NTL) have roughly K times the memory requirement as other deep learning methods (e.g. DSVDD). However, the results in Figure A.2 show that even with small K NTL achieves competitive results.

A.3 Visualization of LNT Detection

Visualizations of LNT detection on LibriSpeech data are reported in Figure A.3. These plots show samples from the test set with artificial anomalies placed at the yellow shaded area in the top row, below $\mathcal{L}_{\text{DDCL}}$ loss yielded by LNT. The reference loss is the output when the uncorrupted sample is fed. To derive a binary decision about the anomaly, instead of thresholding each anomaly score separately, we exploit the sequential nature of the data. We use a downstream Hidden Markov Model with binary states and extract the maximum likelihood state trajectory with Viterbi Decoding. This will smooth the outputs and help detect entire regions that are considered anomalous. From Figure A.3 we can see, that LNT yields higher loss/score values in the corrupted region and can therefore detect the anomalous region successfully.

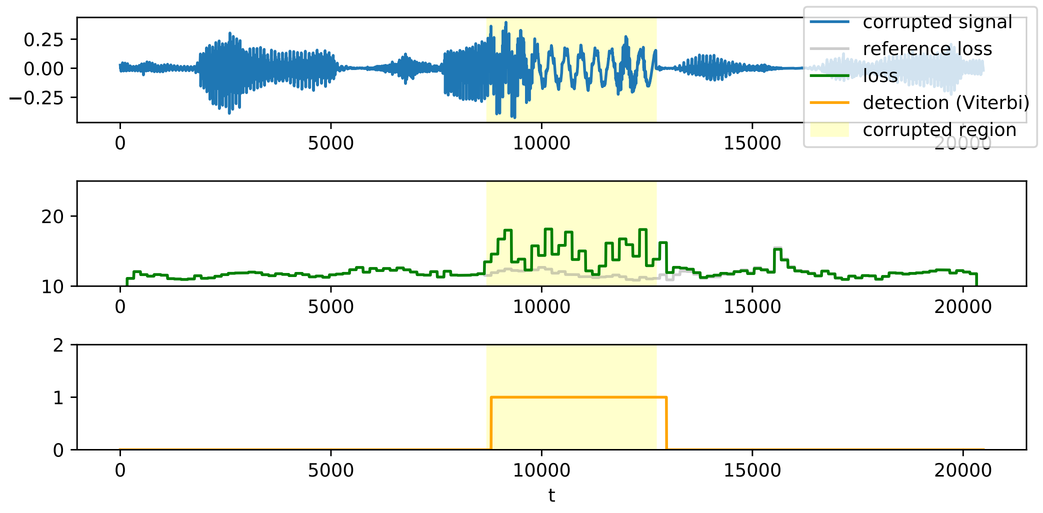


Figure A.3: Top row shows the corrupted signal with an anomaly artificially placed in the highlighted region. This highlight also serves as the ground truth for the last row, which shows the binary decision of the detection. The middle row shows the loss output by LNT for both the corrupted input signal and for the same signal without corruptions (reference loss). The reference loss is only used for visualization purposes to emphasize the locality of LNT.

A.4 Ablation Study of LOE

We review two additional self-supervised anomaly detection methods that are compatible with our approach. They are Multi-Head RotNet (MHRot) [39] for images and Internal Contrastive Learning (ICL) [166] for tabular data.

Multi-Head RotNet (MHRot). MHRot [39] is a self-supervised image anomaly detection method. It learns a multi-head classifier f^θ to predict the applied image transformations including rotation, horizontal shift, and vertical shift. We denote K combined image transformations as $\{\phi_1, ..., \phi_K\}$. The model learns to solve three different tasks: one for predicting rotations, one for predicting vertical shifts, and one for predicting horizontal shifts. The model has three softmax heads, each for a classification task l, modeling the prediction distribution of a transformed image $p^l(\cdot|\phi_k(\boldsymbol{x}); \theta)$. Aiming to predict the correct transformations for normal samples, we maximize the log-likelihoods of the ground truth label t_k^l for each image transformation and each head; for anomalies, we make the predictions evenly distributed by minimizing the cross-entropy from a uniform distribution to the prediction distribution, resulting in

$$\ell_n(\boldsymbol{x}) := -\sum_{k=1}^{K}\sum_{l=1}^{3}\log p^l(t_k^l|\phi_k(\boldsymbol{x});\theta), \quad \ell_a(\boldsymbol{x}) := \sum_{k=1}^{K}\sum_{l=1}^{3}\mathrm{CE}(\mathrm{Unif}, p^l(\cdot|\phi_k(\boldsymbol{x});\theta))$$

We present the experimental results of MHRot on CIFAR10 and FMNIST in table A.1, where we set the contamination ratio $\alpha_0 = \alpha = 0.1$. The results are reported as the mean and standard deviation of three runs with different model initialization and anomaly samples for the contamination. The number in the brackets is the average performance difference from the model trained on clean data. When we use MHRot on raw images, our LOE methods outperform the "Blind" and "Refine" baselines by about 2% AUC on both datasets.

Table A.1: AUC (%) with standard deviation for anomaly detection on CIFAR10 and FMNIST. For all experiments, we set the contamination ratio as 10%. LOE mitigates the performance drop when MHRot trained on the contaminated datasets.

	CIFAR10	FMNIST
Blind	84.0±0.5 (-4.2)	88.8±0.1 (-4.9)
Refine	84.4±0.1 (-3.8)	89.6±0.2 (-4.1)
LOE$_H$ (ours)	**86.4±0.5 (-1.8)**	**91.4±0.2 (-2.3)**
LOE$_S$ (ours)	86.3±0.2 (-1.9)	91.2±0.4 (-2.5)

Internal Contrastive Learning (ICL). ICL [166] is a state-of-the-art *tabular* anomaly detection method. Assuming that the relations between a subset of the features (table columns) and the complementary subset are class-dependent, ICL is able to learn an anomaly detector by discovering the feature relations for a specific class. With this in mind, ICL learns to maximize the mutual information between the two complementary feature subsets, $a(\boldsymbol{x})$ and $b(\boldsymbol{x})$, in the embedding space. The maximization of the mutual information is equivalent to minimizing a contrastive loss on normal samples with two encoders f^θ and g^θ.

$$\ell_n(\boldsymbol{x};\theta) := -\sum_{k=1}^{K}\log p_k(\boldsymbol{x};\theta) \quad \text{with} \quad p_k(\boldsymbol{x};\theta) = \frac{h(f^\theta(a_k(\boldsymbol{x})), g^\theta(b_k(\boldsymbol{x})))}{\sum_{l=1}^{K} h(f^\theta(a_l(\boldsymbol{x})), g^\theta(b_k(\boldsymbol{x}))}.$$

For anomalies, we flip the objective as

$$\ell_a(\boldsymbol{x};\theta) := -\sum_{k=1}^{K}\log(1-p_k(\boldsymbol{x};\theta)).$$

We report F1-scores of ICL on 30 tabular datasets in table A.2. The results are reported as the mean and standard derivation of five runs with different model initializations and random training set split. We set the contamination ratio $\alpha_0 = \alpha = 0.1$ for all datasets. With the backbone model ICL, LOE also outperforms the “Blind" and “Refine" baselines consistently.

Table A.2: F1-score (%) of ICL for anomaly detection on 30 tabular datasets studied in Shenkar and Wolf [166]. We set $\alpha = \alpha_0 = 10\%$ in all experiments. The number in the brackets is the average performance difference from the model trained on clean data. LOE (proposed) outperforms the “Blind” and “Refine” consistently.

	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)
abalone	50.9±1.5(-11.2)	**54.3±2.9(-7.8)**	53.4±5.2(-8.7)	51.7±2.4(-10.4)
annthyroid	29.1±2.2(-12.0)	38.5±2.1(-2.6)	**48.7±7.6(+7.6)**	43.0±8.8(+1.9)
arrhythmia	53.9±0.7(-7.6)	60.9±2.2(-0.6)	62.4±1.8(+0.9)	**63.6±2.1(+2.1)**
breastw	92.6±1.1(-2.4)	93.4±1.0(-1.6)	**96.0±0.6(+1.0)**	95.7±0.6(+0.7)
cardio	50.2±4.5(-19.5)	56.2±3.4(-13.5)	**71.1±3.2(+1.4)**	62.2±2.7(-7.5)
ecoli	17.8±15.1(-55.5)	46.7±25.7(-26.6)	**75.6±4.4(+2.3)**	75.6±4.4(+2.3)
forest	9.2±4.5(-37.8)	8.0±3.6(-39.0)	6.8±3.6(-40.2)	**11.1±2.1(-35.9)**
glass	8.9±4.4(-13.3)	**11.1±0.0(-11.1)**	11.1±7.0(-11.1)	8.9±8.3(-13.3)
ionosphere	86.5±1.1(-5.7)	85.9±2.3(-6.3)	85.7±2.8(-6.5)	**88.6±0.6(-3.6)**
kdd	99.3±0.1(-0.1)	99.4±0.1(+0.0)	**99.5±0.0(+0.1)**	99.4±0.0(+0.0)
kddrev	97.9±0.5(-0.9)	98.4±0.4(-0.4)	**98.8±0.1(+0.0)**	98.2±0.4(-0.6)
letter	43.0±2.5(-15.5)	51.2±3.7(-7.3)	**54.4±5.6(-4.1)**	47.2±4.9(-11.3)
lympho	43.3±8.2(-40.0)	60.0±8.2(-23.3)	80.0±12.5(-3.3)	**83.3±10.5(+0.0)**
mammo.	8.8±1.9(-14.0)	11.4±1.9(-11.4)	34.0±20.2(+11.2)	**42.8±17.6(+20.0)**
mnist	72.1±1.0(-10.5)	80.7±0.7(-1.9)	**86.0±0.4(+3.4)**	79.2±0.9(-3.4)
mulcross	70.4±13.4(-29.6)	94.4±6.3(-5.6)	**100.0±0.0(+0.0)**	99.9±0.1(-0.1)
musk	6.2±3.0(-93.8)	**100.0±0.0(+0.0)**	100.0±0.0(+0.0)	100.0±0.0(+0.0)
optdigits	0.8±0.5(-62.4)	**1.3±1.1(-61.9)**	1.2±1.0(-62.0)	0.9±0.5(-62.3)
pendigits	10.3±4.6(-67.9)	30.1±8.5(-48.1)	80.3±6.1(+2.1)	**88.6±2.2(+10.4)**
pima	58.1±2.9(-2.2)	59.3±1.4(-1.0)	**63.0±1.0(+2.7)**	60.1±1.4(-0.2)
satellite	72.7±1.3(-2.1)	72.7±0.6(-2.1)	**73.6±0.2(-1.2)**	73.2±0.6(-1.6)
satimage	7.3±0.6(-82.0)	85.1±1.4(-4.2)	91.3±1.1(+2.0)	**91.5±0.9(+2.2)**
seismic	14.9±1.4(-3.0)	17.3±2.1(-0.6)	23.6±2.8(+5.7)	**24.2±1.4(+6.3)**
shuttle	96.6±0.2(-0.4)	96.7±0.1(-0.3)	96.9±0.1(-0.1)	**97.0±0.2(+0.0)**
speech	0.3±0.7(-4.1)	1.6±1.0(-2.8)	**2.0±0.7(-2.4)**	0.7±0.8(-3.7)
thyroid	45.8±7.3(-31.4)	71.6±2.4(-5.6)	**83.2±2.9(+6.0)**	80.9±2.5(+3.7)
vertebral	8.9±3.1(-7.8)	8.9±4.2(-7.8)	7.8±4.2(-8.9)	**10.0±2.7(-6.7)**
vowels	42.1±9.0(-37.5)	60.4±7.9(-19.2)	**81.6±2.9(+2.0)**	74.4±8.0(-5.2)
wbc	50.5±5.7(-8.2)	50.5±2.3(-8.2)	**61.0±4.7(+2.3)**	61.0±1.9(+2.3)
wine	4.0±4.9(-86.0)	10.0±8.9(-80.0)	98.0±4.0(+8.0)	**100.0±0.0(+10.0)**

A.5 Ablation Study of ALOE

We investigate the general applicability of ALOE to different backbone models and the benefit of diverse querying.

A.5.1 Study of Backbone Models

We are interested in whether ALOE works for different backbone models. To that end, we repeat part of the experiments in Table 4.6 but using an self-supervised learning model MHRot [39] and a OCC model DSVDD [89] as the backbone model. We compare ALOE to two best performing baselines — Hybr1 and Hybr2. In this experiment, MHRot and DSVDD take different input types: while MHRot takes raw images as input, DSVDD uses pre-trained image features. We also set the query budget to be $K = 20$.

We report the results in Table A.3. It showcases the superiority of ALOE compared to the baselines. On all datasets, ALOE significantly outperforms the two best performing baselines, Hybr1 and Hybr2, thus demonstrating the wide applicability of ALOE across anomaly detection model types.

Table A.3: AUC (%) with standard deviation for anomaly detection on six datasets (CIFAR10, FMNIST, Blood, OrganA, OrganC, OrganS). The backbone models are MHRot [39] and DSVDD [89]. For all experiments, we set the contamination ratio as 10% and query 20 samples. ALOE consistently outperforms two best-performing baselines on all six datasets.

	MHRot			DSVDD		
	ALOE	Hybr1	Hybr2	ALOE	Hybr1	Hybr2
CIFAR10	**86.9**±**0.7**	83.9±0.1	49.1±2.0	**93.1**±**0.2**	89.0±0.6	91.3±1.0
FMNIST	**92.6**±**0.1**	87.1±0.2	58.9±5.7	**91.4**±**0.5**	90.9±0.4	82.5±2.9
Blood	**83.3**±**0.2**	81.1±2.5	61.8±2.1	**80.2**±**1.1**	79.7±1.2	77.2±3.0
OrganA	**96.5**±**0.3**	94.1±0.3	61.1±4.8	**89.5**±**0.3**	87.1±0.7	71.3±3.8
OrganC	**92.1**±**0.2**	91.6±0.1	70.9±0.8	**87.5**±**0.7**	85.3±0.8	84.2±0.9
OrganS	**89.3**±**0.2**	88.3±0.3	68.2±0.1	**85.5**±**0.7**	83.4±0.3	81.2±1.3

A.5.2 Study of Querying Strategies

To understand the benefit of sampling diverse queries with K-means++ , we run the following experiment: We train various anomaly detectors using a supervised loss only on the queried samples. The only difference between them is just the querying

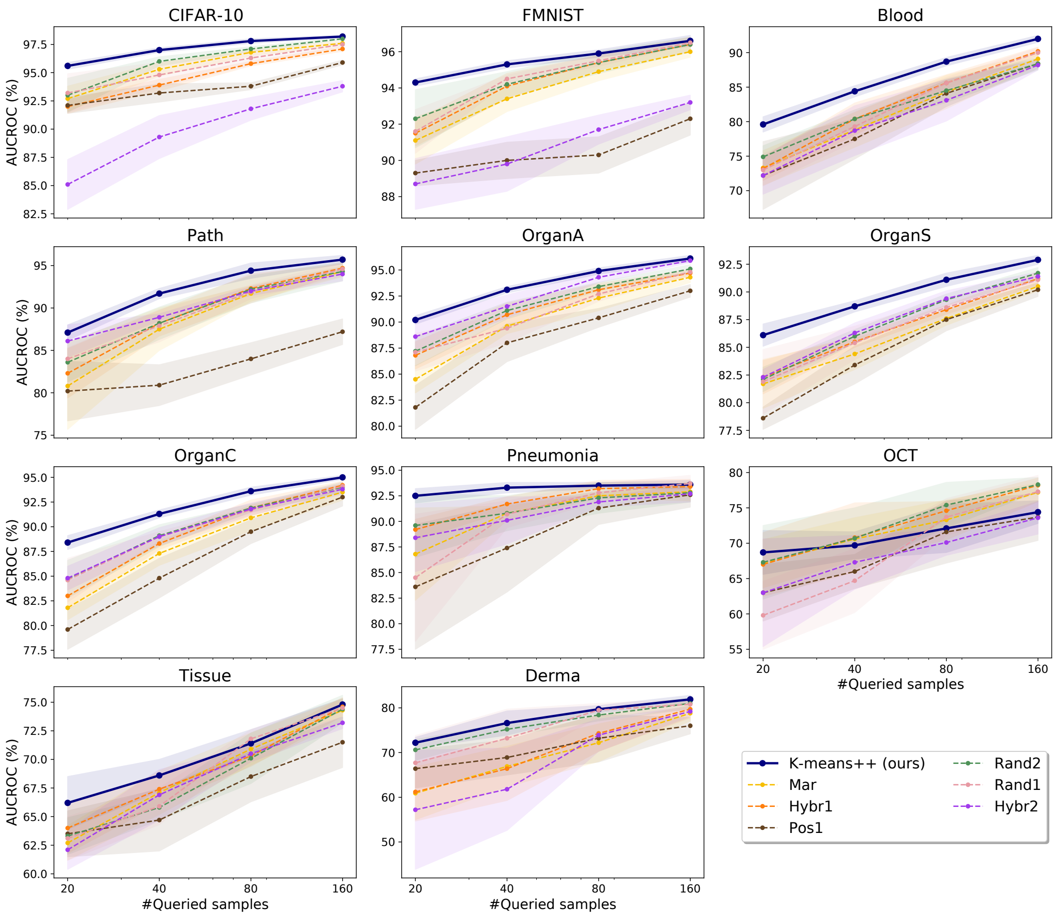


Figure A.4: Ablation study on the query strategy. K-Means++ significantly outperforms other strategies for active anomaly detection on most of the datasets.

strategy used to select the samples. We evaluate them on all image datasets we studied for query budget $|\mathcal{Q}| = 20, 40, 80, 160$.

The results are reported in Figure A.4 in terms of AUC (%). On all datasets except OCT, K-means++ consistently outperforms all other querying strategies from previous works on active anomaly detection. The difference is particularly large when only a few samples are queried.

B Supplementary Details

B.1 Details of NTL Experiments

B.1.1 NTL Implementation on Time Series

The networks in the neural transformations used in all time series experiments consist of one 1D convolutional layer on the bottom, a stack of three residual blocks of 1D convolutional layers with affine-free instance normalization layers, and ReLU activations, as well as one 1D convolutional layer on the top. All convolutional layers are with the kernel size of 3, and the stride of 1. All bias terms are fixed as zero. The dimension of the residual blocks is the data dimension. The convolutional layer on the top has an output dimension as the data dimension. For the multiplicative parameterization, a sigmoid activation is added to the end.

The encoder used in all experiments consists of residual blocks of 1D convolutional layers with ReLU activations, as well as one 1D convolutional layer on the top of all residual blocks.The detailed network structure (from bottom to top) in each time series dataset is:

- SAD: (i) one residual block with the kernel size of 3, the stride of 1, and the output dimension of 32. (ii) four residual blocks with the kernel size of 3, the stride of 2, and the output dimensions of 32, 64, 128, and 256. (iii) one 1D convolutional layer with the kernel size of 6, the stride of 1, and the output dimension of 32.

- NATOPS: (i) one residual block with the kernel size of 3, the stride of 1, and the output dimension of 32. (ii) four residual blocks with the kernel size of 3, the stride of 2, and the output dimensions of 32, 64, 128, and 256. (iii) one 1D convolutional layer with the kernel size of 4, the stride of 1, and the output dimension of 64.

- CT: (i) one residual block with the kernel size of 3, the stride of 1, and the output dimension of 32. (ii) six residual blocks with the kernel size of 3, the stride of 2, and the output dimensions of 32, 64, 128, 256, 512, and 1024. (iii) one 1D convolutional layer with the kernel size of 3, the stride of 1, and the output dimension of 64.

- EPSY: (i) one residual block with the kernel size of 3, the stride of 1, and the output dimension of 32. (ii) six residual blocks with the kernel size of 3, the stride of 2, and the output dimensions of 32, 64, 128, 256, 512, and 1024. (iii) one 1D convolutional layer with the kernel size of 4, the stride of 1, and the output dimension of 128.

- RS: (i) one residual block with the kernel size of 3, the stride of 1, and the output dimension of 32. (ii) three residual blocks with the kernel size of 3, the stride of 2, and the output dimensions of 32, 64, and 128. (iii) one 1D convolutional layer with the kernel size of 4, the stride of 1, and the output dimension of 64.

B.1.2 Time Series Anomaly Detection Baselines

To study the effectiveness of NTL on time series, we implement the following unsupervised and self-supervised anomaly detection time series baselines.

- Deep Anomaly Detection Baselines. The implementations of DSVDD, DROCC, and DAGMM are adopted from the published codes with a similar encoder as NTL. DAGMM has a hyperparameter of the number of mixture components. We consider the number of components between 4 and 12 and select the best-performing one.

- Self-supervised Anomaly Detection Baselines. The implementation of GOAD is taken from the published code. The results of GOAD depend on the choice of the output dimension r of affine transformations. We consider the reduced dimension $r \in \{2^2, 2^3, ..., 2^6\}$, and select the best performing one. We craft specific time series transformations for the designed classification-based baseline. The hand-crafted transformations are the compositions of flipping along the time axis (true/false), flipping along the channel axis (true/false), and shifting along the time axis by 0.25 of its time length (forward/backward/none). By taking all possible compositions, we obtain a total of $2 * 2 * 3 = 12$ transformations.

- Anomaly Detection Baselines for Time Series. The RNN is parameterized by two layers of recurrent neural networks, e.g., GRU, and a stack of two linear layers with ReLU activation on top of it which outputs the mean and variance at each time step. The implementation of LSTM-ED is taken from the web.

B.2 Details of Graph-level Anomaly Detection Experiments

B.2.1 Datasets Statistics

We select six graph classification datasets for the evaluation from three domains (bioinformatics, molecules, and social networks). For each dataset, we report the number of graphs, the dimension of node attributes, the average number of nodes, and the average number of edges in each class as statistics. We list the statistics of these six datasets in Table B.1.

Table B.1: The statistics of used datasets. We report the number of graphs, the dimension of node attributes, the average number of nodes, and the average number of edges for each class in each dataset.

Dataset	Category	Class	#Graphs	#NodeAttrs	Avg.#Nodes	Avg.#Edges
DD	Bioinformatics	0	691	89	355.2	1806.6
		1	487	89	183.7	898.9
PRTOEINS	Bioinformatics	0	663	3	50.0	188.1
		1	450	3	22.9	83.0
NCI1	Molecules	0	2053	37	25.7	55.3
		1	2057	37	34.1	73.9
AIDS	Molecules	0	400	38	37.6	80.5
		1	1600	38	10.2	20.4
IMDB-B	Social networks	0	500	136	20.1	193.6
		1	500	136	19.4	192.6
REDDIT-B	Social networks	0	1000	1	641.3	1471.9
		1	1000	1	218.0	519.1

⋆ In IMDB-B, the one-hot degree is used as node attributes. In REDDIT-B, the constant one is used as node attributes.

B.2.2 Baselines Details

To build a comprehensive benchmark for graph-level anomaly detection, we include both GNN-based methods and non-GNN-based methods as baselines.

Graph Transformation Prediction (GTP) is a self-supervised detection method based on transformation prediction. It is an end-to-end detection method optimizing the classifier and scoring the anomalies using the cross-entropy of predictions and ground truth labels. We use six graph transformations (including the identity transformation), which are originally designed for graph contrastive learning in You et al. [149]. For completeness, we list their details below.

- Node dropping: it randomly discards 10% of vertices along with their connections given the graph. Each node dropping probability follows a uniform distribution.

- Edge adding: it randomly adds 10% of edges to the graph. Each edge adding probability follows a uniform distribution.

- Edge dropping: it randomly removes 10% of the edges. The dropping probability follows a uniform distribution.

- Attribute masking: given the graph, it firstly selects 10% of nodes randomly and then masks all feature dimensions of the selected node with a Gaussian distribution.

- Subgraph: it samples a subgraph with a ratio of 10% from the graph using a random walk.

Non-GNN based baselines. In the empirical study, we include non-GNN based methods as baselines. They are two-stage detection methods using either unsupervised graph embedding methods (G2V [151] or (FGSD) [152]) or graph kernels (WLK [153] or PK [154]) as the first stage model. Anomalies can be detected by training an OCSVM on the features obtained from the first stage model. Their descriptions are listed below.

- G2V is an unsupervised graph embedding method. It views an entire graph as a document and sub-graphs as words that compose the document and learn the graph-level representation using the doc2vec algorithm.

- FGSD is an unsupervised graph embedding method. It calculates the Moore-Penrose spectrum of the normalized Laplacian and uses it as the graph-level representation.

- WLK is a graph kernel algorithm. It compares graphs in different hierarchical levels by iteratively relabeling graphs using the WL algorithm and constructing a base graph kernel applied at each level.

- PK is a graph kernel algorithm. It propagates node information between nodes based on the graph structure.

The implementations of G2V and FGSD are from Karate Club library [214]. The implementations of WLK and PK are from GraKel [215].

B.2.3 Training Hyperparameters

We use the Adam optimizer [94] with an initial learning rate of 0.001 and decay the learning rate by 0.5 every 100 epochs. We set the maximum epochs as 500 and the batch size as 128. We use the early stopping based on validation loss (without access to the true labeled anomalies) for the training. The early stopping is implemented with a patience parameter of 100 epochs to ease the sensitivity to fluctuations in the validation loss. An early stopping without access to the true anomalies is critical for an unbiased model evaluation in the anomaly detection tasks.

B.3 Details of LOE Experiments

B.3.1 Baseline Details

Across all experiments, we employ two baselines that do not utilize anomalies to help train the models. The baselines are either completely blind to anomalies, or drop the perceived anomalies' information. Normally training a model without recognizing anomalies serves as our first baseline. Since this baseline doesn't take any action to the anomalies in the contaminated training data and is actually blind to the anomalies that exist, we name it *Blind*. Mathematically, Blind sets $y_i = 0$ in Equation (4.1) for all samples.

The second baseline filters out anomalies and refines the training data: at every mini-batch update, it first ranks the mini-batch data according to the anomaly scores given the current detection model, then removes the top α most likely anomalous samples from the mini-batch. The remaining samples perform the model update. We name the second baseline *Refine*, which still follows Alg. 2 but removes ℓ_a in Equation (4.1). Both these two baselines take limited actions to the anomalies. We use them to contrast our proposed methods and highlight the useful information contained in unseen anomalies.

B.3.2 Training Hyperparameters

NTL on image data. On CIFAR10, we set the mini-batch size to be 500, the learning rate to be 4e-4, 30 training epochs with Adam optimizer [94]. On FMNIST, we set the mini-batch size to be 500, the learning rate to be 2e-4, 30 training epochs with the Adam optimizer. On MVTEC, we set the mini-batch size to be 40, the learning rate to be 2e-4, 30 training epochs with Adam optimizer. For the "Refine" baseline and our methods we set the number of warm-up epochs as two on all image datasets.

NTL on tabular data. We use Adam optimizer with a learning rate chosen from $[5e-4, 1e-3, 2e-3]$. For the "Refine" baseline and our methods we set the number of warm-up epochs as two for small datasets and as one for large datasets.

NTL on video data. We use Adam stochastic optimizer with a batch size of 192 and a learning rate of 1e-4. We update the model for 3 epochs and report the results with three independent runs.

B.4 Details of ALOE Experiments

B.4.1 Baselines Details

we describe the details of the baselines in Table 4.5 in Section 4.2.2. For each baseline method, we explain their query strategies and post-query training strategies. Please also refer to our codebase for practical implementation details.

- **Rand1.** This strategy used by Ruff et al. [22] selects queries by sampling uniformly without replacement across the training set, resulting in the queried index set $\mathcal{Q} = \{i_q \sim \text{Unif}(1, \cdots, N) | 1 \leq q \leq |\mathcal{Q}|\}$. After the querying, models are trained with a supervised loss function based on outlier exposure on the labeled data and with a one-class classification loss function on the unlabeled data,

$$\mathcal{L}^{\theta}_{\text{Rand1}} = \frac{1}{|\mathcal{Q}|} \sum_{j \in \mathcal{Q}} (y_j \ell_a(\boldsymbol{x}_j; \theta) + (1 - y_j)\ell_n(\boldsymbol{x}_j; \theta)) + \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \ell_n(\boldsymbol{x}_i; \theta). \tag{B.1}$$

  As in ALOE both loss contributions are weighted equally. $\mathcal{L}^{\theta}_{\text{Rand1}}$ is minimized with respect to the backbone model parameters θ.

- **Rand2.** The querying strategy of Trittenbach et al. [184] samples uniformly among the top 50% data ranked by anomaly scores without replacement. This leads to a random set of "positive" queries. After the queries are labeled, the training loss function is the same as $\mathcal{L}^{\theta}_{\text{Rand1}}$ (Equation (B.1)).

- **Mar.** After training the backbone model for one epoch, this querying strategy by Görnitz et al. [173] uses the α-quantile (s_α) of the training data anomaly scores to define a "normality region". Then the K samples closest to the margin s_α are selected to be queried. After the queries are labeled, the training loss function is the same as $\mathcal{L}^{\theta}_{\text{Rand1}}$ (Equation (B.1)). Note that in practice we don't know the true anomaly ratio for the α-quantile. In all experiments, we provide this querying strategy with the true contamination ratio of the dataset. Even with the true ratio, the "Mar" strategy is still outperformed by ALOE.

- **Hybr1.** This hybrid strategy, also used by [173] combines the "Mar" query with neighborhood-based diversification. The neighborhood-based strategy selects samples with fewer neighbors covered by the queried set to ensure the samples' diversity in the feature space. We start by selecting the data index $\arg\min_{1\leq i\leq N}\|s_i - s_\alpha\|$ into $\mathcal{Q}$. Then the samples are selected sequentially without replacement by the criterion
$$\mathop{\arg\min}_{1\leq i\leq N} 0.5 + \frac{|\{j \in \mathrm{NN}_k(f^\theta(\boldsymbol{x}_i)) : j \in \mathcal{Q}\}|}{2k} + \beta\frac{\|s_i - s_\alpha\| - \min_i\|s_i - s_\alpha\|}{\max_i\|s_i - s_\alpha\| - \min_i\|s_i - s_\alpha\|}$$
where the inter-sample distance is measured in the feature space and the number of nearest neighbors is $k = \lceil N/K\rceil$. We set $\beta = 1$ for an equal contribution of both terms. After the queries are labeled, the training loss function is the same as $\mathcal{L}^\theta_{\mathrm{Rand1}}$ (Equation (B.1)).

- **Pos1.** This querying strategy by Pimentel et al. [177] always selects the top-ranked samples ordered by their anomaly scores, $\arg\max_{1\leq i\leq N} s_i$. After the queries are labeled, the training loss only involves the labeled data
$$\mathcal{L}^\theta_{\mathrm{Pos1}} = \frac{1}{|\mathcal{Q}|}\sum_{j\in\mathcal{Q}}(y_j\ell_a(\boldsymbol{x}_j;\theta) + (1-y_j)\ell_n(\boldsymbol{x}_j;\theta)).$$
Pimentel et al. [177] use the logistic loss but we use the supervised outlier exposure loss. The supervised outlier exposure loss is shown to be better than the logistic loss in learning anomaly detection models [22, 31].

- **Pos2.** This approach of [183] uses the same querying strategy as Pos1, but the training is different. Pos2 also uses the unlabeled data during training. After the queries are labeled, the training loss function is the same as $\mathcal{L}^\theta_{\mathrm{Rand1}}$ (Equation (B.1)).

- **Hybr2.** This hybrid strategy by Das et al. [176] makes positive diverse queries. It combines querying according to anomaly scores with distance-based diversification. Hybr2 selects the initial query $\arg\max_{1\leq i\leq N} s_i$ into $\mathcal{Q}$. Then the samples are selected sequentially without replacement by the criterion
$$\mathop{\arg\max}_{1\leq i\leq N}\frac{s_i - \min_i s_i}{\max_i s_i - \min_i s_i} + \beta\min_{j\in\mathcal{Q}}\frac{d(\boldsymbol{x}_i,\boldsymbol{x}_j) - \min_{a\neq b} d(\boldsymbol{x}_a,\boldsymbol{x}_b)}{\max_{a\neq b} d(\boldsymbol{x}_a,\boldsymbol{x}_b) - \min_{a\neq b} d(\boldsymbol{x}_a,\boldsymbol{x}_b)}$$
where $d(\boldsymbol{x}_i,\boldsymbol{x}_j) = ||f^\theta(\boldsymbol{x}_i) - f^\theta(\boldsymbol{x}_j)||_2$. We set $\beta = 1$ for an equal contribution of both terms. After the queries are labeled, Das et al. [176] use the labeled set to learn a set of weights for the components of an ensemble of detectors. For a fair comparison of active learning strategies, we use the labeled set to update an individual anomaly detector with parameters θ by optimizing the loss
$$\mathcal{L}^\theta_{\mathrm{Hybr2}} = \frac{1}{|\mathcal{Q}|}\sum_{j\in\mathcal{Q}}(y_j\ell_a(\boldsymbol{x}_j;\theta) + (1-y_j)\ell_n(\boldsymbol{x}_j;\theta)).$$

- **Hybr3.** This baseline by [178] uses the same query strategy as Hybr2, but differs in the training loss function,

$$\mathcal{L}^{\theta}_{\text{Hybr3}} = \frac{1}{|\mathcal{Q}|+|\mathcal{U}|}\sum_{j\in\mathcal{Q}} w_j(1-y_j)\ell_n(\boldsymbol{x}_j;\theta) + \frac{1}{|\mathcal{Q}|+|\mathcal{U}|}\sum_{i\in\mathcal{U}} \hat{w}_i\ell_n(\boldsymbol{x}_i;\theta),$$

  where $w_j = 2\sigma(d_j)$ and $\hat{w}_i = 2 - 2\sigma(d_i)$ where $\sigma(\cdot)$ is the sigmoid function and $d_i = \text{minmax}(||f^{\theta}(\boldsymbol{x}_i) - \boldsymbol{c}_0||_2 - ||f^{\theta}(\boldsymbol{x}_i) - \boldsymbol{c}_1||_2)$ where $\boldsymbol{c}_0$ is the center of the queried normal samples and $\boldsymbol{c}_1$ is the center of the queried abnormal samples in the feature space, and minmax is the min-max scaling.

  We make three observations for the loss function. First, $\mathcal{L}^{\theta}_{\text{Hybr3}}$ filters out all labeled anomalies in the supervised learning part and assigns a large weight (but only as large as two at most) to the true normal data that has a high anomaly score. Second, in the unlabeled data, $\mathcal{L}^{\theta}_{\text{Hybr3}}$ assigns a smaller weight (less than 1) to the seemingly abnormal data. Third, overall, the weight of the labeled data is similar to the weight of the unlabeled data. This is unlike ALOE, which weighs labeled data $|\mathcal{U}|/|\mathcal{Q}|$ times higher than unlabeled data.

B.4.2 Training Hyperparameters

In the experiments, we use Adam [94] in the training of NTL to find the local optimal anomaly scorer parameters θ. For Adam, we set $\beta_1 = 0.9, \beta_2 = 0.999$ and no weight decay for all experiments. To set the learning rate, training epochs, and minibatch size for MedMNIST, we find the best performing hyperparameters by evaluating the method on the validation dataset. We use the same hyperparameters on other image data. We summarize all optimization hyperparameters in Table B.2.

Table B.2: A summary of optimization parameters for NTL on image datasets and tabular datasets.

Dataset	Learning Rate	Epoch	Minibatch Size	τ
CIFAR10	1e-4	30	512	1e-2
FMNIST	1e-4	30	512	1e-2
MedMNIST	1e-4	30	512	1e-2
ODDS	1e-3	100	$\lceil N/5 \rceil$	1e-2

List of Figures

List of Tables

Glossary

ALOE Active Latent Outlier Exposure 3, 5, 10, 67, 82, 84–86, 88–93, 96, 109, 116, 118, 121, 124, 125

Contra-DA Distribution-Augmented Contrastive Learning 33, 34

CPC Contrastive Predictive Coding 43–50, 53, 97, 120

CT Character Trajectories 25–27, 30, 111

CVDD Context Vector Data Description 40, 41

DAGMM Deep Autoencoding Gaussian Mixture Model 26, 27, 30–32, 112

DCL Deterministic Contrastive Loss 11–14, 18–22, 42, 56, 104

DDCL Dynamic Deterministic Contrastive Loss 45–49, 53, 120

DMSVDD Deep Multi-sphere Support Vector Data Description 40, 41

DN2 Deep Nearest-Neighbors 33, 34, 36

DROCC Deep Robust One-Class Classification 31, 32, 112

DSVDD Deep Support Vector Data Description 26, 27, 30–34, 37, 40, 41, 57, 69, 73, 74, 105, 109, 112, 120, 125

EPSY Epilepsy 25–27, 30, 112

FCDD Fully Convolutional Data Description 37, 38

FGSD 114

G2V Graph2Vec 61–63, 114

GEOM Geometric Transformation Classification 33, 34, 36

GIN Graph Isomorphism Network 61

GNN Graph Neural Network 55, 56, 58, 61, 62, 64, 65, 113, 114, 120

GOAD 26, 27, 30–32, 112

GTL Graph Transformation Learning 55, 56, 58, 61–63

GTP Graph Transformation Prediction 61–64

ICL Internal Contrastive Learning 106–108, 125

IF Isolation Forest 1, 26, 27, 30–32

KDE Kernel Density Estimation 1, 7, 9, 39, 81

KNN K-Nearest Neighbor 26, 29, 32–34, 36, 39, 97, 119

LNT Local Neural Transformations 2, 5, 43–53, 65, 95, 105, 106, 120, 121, 123

LOE Latent Outlier Exposure 3, 5, 10, 67–84, 92, 93, 96, 107, 108, 120, 121, 124, 125

LOF Local Outlier Factor 1, 26, 27, 29–32, 36, 119

LSTM-ED LSTM-based Encoder-Decoder 26, 27, 30, 50, 51, 112

MHRot Multi-Head RotNet 106, 107, 109, 125

MOCC Multi-view One-Class Classification 3, 5, 43, 54, 58–66, 95, 96, 124

MSCRED Multi-Scale Convolutional Recurrent Encoder-Decoder 50, 51

mSVDD Multi-modal Deep Support Vector Data Description 40, 41

NATOPS Naval Air Training and Operating Procedures Standardization 25–27, 29, 30, 104, 105, 111, 119, 121

NTL Neural Transformation Learning 2–5, 10–12, 14–16, 21–44, 59, 65, 67, 68, 72, 74–78, 80, 82, 87, 88, 95, 96, 103–105, 112, 115, 116, 118–121, 123–125

OCC One-Class Classification 43, 54–59, 63, 65, 66, 82, 95, 96, 99, 109

OCGIN One-Class GIN 54, 57, 61–63, 65

OCGTL One-Class Graph Transformation Learning 3, 5, 43, 54, 55, 57, 58, 60–66, 95, 96, 124

OCPool One-Class Pooling 62–64, 120

OCSVM One-Class Support Vector Machine 1, 7, 9, 17, 26, 27, 30–33, 40, 41, 53, 61, 62, 81, 97, 114

PaDIM Patch Distribution Modeling 38

PANDA Pretrained Anomaly Detection Adaptation 33–35, 123

PCA Principal Component Analysis 1

PK Propagation Kernel 61–63, 114

RNN RNN-based Model 26, 27, 30, 112

RS Racket Sports 25–27, 30, 112

SAD Spoken Arabic Digits 25–30, 111, 119

SPADE Semantic Pyramid Anomaly Detection 37, 38

SWaT Secure Water Treatment dataset 49–53, 120, 123

THOC Temporal Hierarchical One-Class Network 50, 51

WLK Weisfeiler-Leman subtree Kernel 61–63, 114

Bibliography

[1] Richard A Kemmerer and Giovanni Vigna. Intrusion detection: a brief history and overview. *Computer*, 35(4):supl27–supl30, 2002.

[2] Jiong Zhang and Mohammad Zulkernine. Anomaly based network intrusion detection with unsupervised outlier detection. In *2006 IEEE International Conference on Communications*, volume 5, pages 2388–2393. IEEE, 2006.

[3] Leman Akoglu and Christos Faloutsos. Anomaly, event, and fraud detection in large network datasets. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 773–774, 2013.

[4] Mohiuddin Ahmed, Abdun Naser Mahmood, and Md Rafiqul Islam. A survey of anomaly detection techniques in financial domain. *Future Generation Computer Systems*, 55:278–288, 2016.

[5] Leo H Chiang, Evan L Russell, and Richard D Braatz. *Fault detection and diagnosis in industrial systems*. Springer Science & Business Media, 2000.

[6] Rahat Iqbal, Tomasz Maniak, Faiyaz Doctor, and Charalampos Karyotis. Fault detection and isolation in industrial processes using deep learning approaches. *IEEE Transactions on Industrial Informatics*, 15(5):3077–3084, 2019.

[7] Arijit Ukil, Soma Bandyoapdhyay, Chetanya Puri, and Arpan Pal. Iot healthcare analytics: The importance of anomaly detection. In *2016 IEEE 30th international conference on advanced information networking and applications (AINA)*, pages 994–997. IEEE, 2016.

[8] Stephan Dreiseitl, Melanie Osl, Christian Scheibböck, and Michael Binder. Outlier detection with one-class svms: an application to melanoma prognosis. In *AMIA annual symposium proceedings*, volume 2010, page 172. American Medical Informatics Association, 2010.

[9] Donald A Jackson and Yong Chen. Robust principal component analysis and outlier detection with ecological data. *Environmetrics: The official journal of the International Environmetrics Society*, 15(2):129–139, 2004.

[10] Tao Chen, Elaine Martin, and Gary Montague. Robust probabilistic pca with missing data and contribution analysis for outlier detection. *Computational Statistics & Data Analysis*, 53(10):3706–3716, 2009.

[11] Shohei Hido, Yuta Tsuboi, Hisashi Kashima, Masashi Sugiyama, and Takafumi Kanamori. Statistical outlier detection using direct density ratio estimation. *Knowledge and information systems*, 26(2):309–336, 2011.

[12] Bo Tang and Haibo He. A local density-based approach for outlier detection. *Neurocomputing*, 241:171–180, 2017.

[13] Larry M Manevitz and Malik Yousef. One-class svms for document classification. *Journal of machine Learning research*, 2(Dec):139–154, 2001.

[14] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[15] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.

[16] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6 (1):1–39, 2012.

[17] Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 2021.

[18] Emanuele Principi, Fabio Vesperini, Stefano Squartini, and Francesco Piazza. Acoustic novelty detection with adversarial autoencoders. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 3324–3330. IEEE, 2017.

[19] Chong Zhou and Randy C Paffenroth. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 665–674, 2017.

[20] Xiaoran Chen and Ender Konukoglu. Unsupervised detection of lesions in brain mri using constrained adversarial auto-encoders. In *MIDL Conference book*. MIDL, 2018.

[21] Sarah M Erfani, Sutharshan Rajasegarar, Shanika Karunasekera, and Christopher Leckie. High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning. *Pattern Recognition*, 58:121–134, 2016.

[22] Lukas Ruff, Robert A Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. Deep semi-supervised anomaly detection. In *International Conference on Learning Representations*, 2019.

[23] Lukas Ruff, Yury Zemlyanskiy, Robert Vandermeulen, Thomas Schnake, and Marius Kloft. Self-attentive, multi-context one-class classification for unsupervised anomaly detection on text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4061–4071, 2019.

[24] Yanshan Xiao, Bo Liu, Longbing Cao, Xindong Wu, Chengqi Zhang, Zhifeng Hao, Fengzhao Yang, and Jie Cao. Multi-sphere support vector data description for outliers detection on multi-distribution data. In *2009 IEEE international conference on data mining workshops*, pages 82–87. IEEE, 2009.

[25] Zahra Ghafoori and Christopher Leckie. Deep multi-sphere support vector data description. In *Proceedings of the 2020 SIAM International Conference on Data Mining*, pages 109–117. SIAM, 2020.

[26] Chenlong Hu, Yukun Feng, Hidetaka Kamigaito, Hiroya Takamura, and Manabu Okumura. One-class text classification with multi-modal deep support vector data description. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3378–3390, 2021.

[27] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging*, pages 146–157. Springer, 2017.

[28] Lucas Deecke, Robert Vandermeulen, Lukas Ruff, Stephan Mandt, and Marius Kloft. Image anomaly detection with generative adversarial networks. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 3–17. Springer, 2018.

[29] Liron Bergman, Niv Cohen, and Yedid Hoshen. Deep nearest neighbor anomaly detection. *arXiv preprint arXiv:2002.10445*, 2020.

[30] Tal Reiss, Niv Cohen, Liron Bergman, and Yedid Hoshen. Panda: Adapting pretrained features for anomaly detection and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2806–2814, 2021.

[31] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *International Conference on Learning Representations*, 2018.

[32] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[33] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9): 2352–2449, 2017.

[34] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

[35] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75, 2018.

[36] Charu C Aggarwal et al. Neural networks and deep learning. *Springer*, 10: 978–3, 2018.

[37] Kihyuk Sohn, Chun-Liang Li, Jinsung Yoon, Minho Jin, and Tomas Pfister. Learning and evaluating representations for deep one-class classification. In *International Conference on Learning Representations*, 2021.

[38] Izhak Golan and Ran El-Yaniv. Deep anomaly detection using geometric transformations. In *Advances in Neural Information Processing Systems*, pages 9758–9769, 2018.

[39] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. In *Advances in Neural Information Processing Systems*, pages 15663–15674, 2019.

[40] Siqi Wang, Yijie Zeng, Xinwang Liu, En Zhu, Jianping Yin, Chuanfu Xu, and Marius Kloft. Effective end-to-end unsupervised outlier detection via

inlier priority of discriminative network. In *Advances in Neural Information Processing Systems*, pages 5962–5975, 2019.

[41] Liron Bergman and Yedid Hoshen. Classification-based anomaly detection for general data. In *International Conference on Learning Representations*, 2020.

[42] Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. In *34th Conference on Neural Information Processing Systems (NeurIPS) 2020*. Neural Information Processing Systems, 2020.

[43] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):4037–4058, 2020.

[44] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018.

[45] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015.

[46] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

[47] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[48] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pages 69–84. Springer, 2016.

[49] Dahun Kim, Donghyeon Cho, Donggeun Yoo, and In So Kweon. Learning image representations by completing damaged jigsaw puzzles. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 793–802. IEEE, 2018.

[50] Richard Zhang, Phillip Isola, and Alexei A Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1058–1067, 2017.

[51] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European conference on computer vision*, pages 527–544. Springer, 2016.

[52] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE international conference on computer vision*, pages 667–676, 2017.

[53] Donglai Wei, Joseph J Lim, Andrew Zisserman, and William T Freeman. Learning and using the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8052–8060, 2018.

[54] Dejing Xu, Jun Xiao, Zhou Zhao, Jian Shao, Di Xie, and Yueting Zhuang. Self-supervised spatiotemporal learning via video clip order prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10334–10343, 2019.

[55] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[56] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2019.

[57] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020.

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[59] Yaochen Xie, Zhao Xu, Jingtun Zhang, Zhengyang Wang, and Shuiwang Ji. Self-supervised learning of graph neural networks: A unified review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[60] Ralph Linsker. Self-organization in a perceptual network. *Computer*, 21(3): 105–117, 1988.

[61] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representa-

tions by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2018.

[62] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[63] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. In *Advances in Neural Information Processing Systems*, pages 4650–4661, 2019.

[64] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. In *International Conference on Learning Representations*, 2020.

[65] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019.

[66] Michael Tschannen, Josip Djolonga, Paul K Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. In *International Conference on Learning Representations*, 2019.

[67] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.

[68] Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(2), 2012.

[69] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[70] Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. *Advances in neural information processing systems*, 26:2265–2273, 2013.

[71] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1734–1747, 2015.

[72] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[73] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[74] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[75] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

[76] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33: 9912–9924, 2020.

[77] Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tieyan Liu. Incorporating bert into neural machine translation. In *International Conference on Learning Representations*, 2019.

[78] Chengwei Chen, Yuan Xie, Shaohui Lin, Ruizhi Qiao, Jian Zhou, Xin Tan, Yi Zhang, and Lizhuang Ma. Novelty detection via contrastive learning with negative data augmentation. *arXiv preprint arXiv:2106.09958*, 2021.

[79] Jouwon Song, Kyeongbo Kong, Ye-In Park, Seong-Gyun Kim, and Suk-Ju Kang. Anoseg: Anomaly segmentation network using self-supervised learning. *arXiv preprint arXiv:2110.03396*, 2021.

[80] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Draem-a discriminatively trained reconstruction embedding for surface anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8330–8339, 2021.

[81] Sachin Goyal, Aditi Raghunathan, Moksh Jain, Harsha Vardhan Simhadri, and Prateek Jain. Drocc: Deep robust one-class classification. In *International Conference on Machine Learning*, pages 3711–3721. PMLR, 2020.

[82] Shohreh Deldari, Daniel V Smith, Hao Xue, and Flora D Salim. Time series change point detection with self-supervised contrastive predictive coding. In *Proceedings of the Web Conference 2021*, pages 3124–3135, 2021.

[83] Chris U Carmona, François-Xavier Aubet, Valentin Flunkert, and Jan Gasthaus. Neural contextual anomaly detection for time series. *arXiv preprint arXiv:2107.07702*, 2021.

[84] Jiaxin Zhang, Kyle Saleeby, Thomas Feldhausen, Sirui Bi, Alex Plotkowski, and David Womble. Self-supervised anomaly detection via neural autoregressive flows with active learning. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[85] Vikash Sehwag, Mung Chiang, and Prateek Mittal. Ssd: A unified framework for self-supervised outlier detection. In *International Conference on Learning Representations*, 2020.

[86] Hyunsoo Cho, Jinseok Seol, and Sang-goo Lee. Masked contrastive learning for anomaly detection. *arXiv preprint arXiv:2105.08793*, 2021.

[87] Tal Reiss and Yedid Hoshen. Mean-shifted contrastive loss for anomaly detection. *arXiv preprint arXiv:2106.03844*, 2021.

[88] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9664–9674, 2021.

[89] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402, 2018.

[90] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020.

[91] Ning Huyan, Dou Quan, Xiangrong Zhang, Xuefeng Liang, Jocelyn Chanussot, and Licheng Jiao. Unsupervised outlier detection using memory and contrastive learning. *arXiv preprint arXiv:2107.12642*, 2021.

[92] Jonathan Muzikar. Installation view of the exhibition "andy warhol: Campbell's soup cans and other works, 1953-1967", 2015. URL `https://www.moma.org/calendar/exhibitions/1517?installation_image_index=29`. April 25, 2015–October 12, 2015. IN2323.30.

[93] Klaus-Robert Müller, Sebastian Mika, Koji Tsuda, and Koji Schölkopf. An introduction to kernel-based learning algorithms. In *Handbook of Neural Network Signal Processing*, pages 4–1. CRC Press, 2018.

[94] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[95] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Asian conference on computer vision*, pages 622–637. Springer, 2018.

[96] Davide Abati, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Latent space autoregression for novelty detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 481–490, 2019.

[97] Pramuditha Perera, Ramesh Nallapati, and Bing Xiang. Ocgan: One-class novelty detection using gans with constrained latent representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2898–2906, 2019.

[98] Jingjing Wang, Sun Sun, and Yaoliang Yu. Multivariate triangular quantile maps for novelty detection. In *Advances in Neural Information Processing Systems*, pages 5060–5071, 2019.

[99] Ki Hyun Kim, Sangwoo Shim, Yongsub Lim, Jongseob Jeon, Jeongwoo Choi, Byungchan Kim, and Andre S Yoon. Rapp: Novelty detection with reconstruction along projection pathway. In *International Conference on Learning Representations*, 2019.

[100] Faruk Ahmed and Aaron Courville. Detecting semantic anomalies. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3154–3162, 2020.

[101] Lucas Deecke, Lukas Ruff, Robert A Vandermeulen, and Hakan Bilen. Transfer-based semantic anomaly detection. In *International Conference on Machine Learning*, pages 2546–2558. PMLR, 2021.

[102] Alejandro Pasos Ruiz, Michael Flynn, James Large, Matthew Middlehurst, and Anthony Bagnall. The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, pages 1–49, 2020.

[103] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The uea multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

[104] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*, 2018.

[105] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Lstm-based encoder-decoder for multi-sensor anomaly detection. *arXiv preprint arXiv:1607.00148*, 2016.

[106] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5188–5196, 2015.

[107] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9592–9600, 2019.

[108] Jihun Yi and Sungroh Yoon. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

[109] Philipp Liznerski, Lukas Ruff, Robert A Vandermeulen, Billy Joe Franks, Marius Kloft, and Klaus Robert Muller. Explainable deep one-class classification. In *International Conference on Learning Representations*, 2020.

[110] Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *arXiv preprint arXiv:2005.02357*, 2020.

[111] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: a patch distribution modeling framework for anomaly detection and localization. In *ICPR 2020-25th International Conference on Pattern Recognition Workshops and Challenges*, 2021.

[112] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL `http://www.aclweb.org/anthology/D14-1162`.

[113] Alexander J Ratner, Henry R Ehrenberg, Zeshan Hussain, Jared Dunnmon, and Christopher Ré. Learning to compose domain-specific transformations for data augmentation. *Advances in neural information processing systems*, 30: 3239, 2017.

[114] Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[115] Xinyu Zhang, Qiang Wang, Jian Zhang, and Zhao Zhong. Adversarial autoaugment. In *International Conference on Learning Representations*, 2019.

[116] Daniel Ho, Eric Liang, Xi Chen, Ion Stoica, and Pieter Abbeel. Population based augmentation: Efficient learning of augmentation policy schedules. In *International Conference on Machine Learning*, pages 2731–2741. PMLR, 2019.

[117] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[118] Toan Tran, Trung Pham, Gustavo Carneiro, Lyle Palmer, and Ian Reid. A bayesian data augmentation approach for learning deep models. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 2794–2803, 2017.

[119] Eric Wong and J Zico Kolter. Learning perturbation sets for robust machine learning. *arXiv preprint arXiv:2007.08450*, 2020.

[120] Alex Tamkin, Mike Wu, and Noah Goodman. Viewmaker networks: Learning views for unsupervised representation learning. In *International Conference on Learning Representations*, 2021.

[121] Chen Qiu, Timo Pfrommer, Marius Kloft, Stephan Mandt, and Maja Rudolph. Neural transformation learning for deep anomaly detection beyond images. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8703–8714. PMLR, 18–24 Jul 2021.

[122] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. In *Proceedings*, volume 89, pages 89–94, 2015.

[123] Pavel Filonov, Andrey Lavrentyev, and Artem Vorontsov. Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model. *arXiv preprint arXiv:1612.06676*, 2016.

[124] Yangdong He and Jiabao Zhao. Temporal convolutional networks for anomaly detection in time series. In *Journal of Physics: Conference Series*, volume 1213, page 042050. IOP Publishing, 2019. doi: 10.1088/1742-6596/1213/4/042050.

[125] Mohsin Munir, Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. Deepant: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access*, 7:1991–2005, 2019. doi: 10.1109/ACCESS.2018.2886457.

[126] Markus Thill, Wolfgang Konen, and Thomas Bäck. Time series encodings with temporal convolutional networks. In *International Conference on Bioinspired Methods and Their Applications*, pages 161–173. Springer, 2020. ISBN 978-3-030-63710-1. doi: 10.1007/978-3-030-63710-1_13.

[127] Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V Chawla. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1409–1416, 2019. doi: 10.1609/aaai.v33i01.33011409.

[128] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. Usad: Unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3395–3404, 2020. doi: 10.1145/3394486.3403392.

[129] Maximilian Sölch, Justin Bayer, Marvin Ludersdorfer, and Patrick van der Smagt. Variational inference for on-line anomaly detection in high-dimensional time series. *arXiv preprint*, 2016.

[130] Daehyung Park, Yuuna Hoshi, and Charles C Kemp. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *IEEE Robotics and Automation Letters*, 3(3):1544–1551, 2018.

[131] Yifan Guo, Weixian Liao, Qianlong Wang, Lixing Yu, Tianxi Ji, and Pan Li. Multidimensional time series anomaly detection: A gru-based gaussian mixture variational autoencoder approach. In *Asian Conference on Machine Learning*, pages 97–112. PMLR, 2018. URL `http://proceedings.mlr.press/v95/guo18a.html`.

[132] Joao Pereira and Margarida Silveira. Unsupervised anomaly detection in energy time series data using variational recurrent autoencoders with attention. In *2018 17th IEEE international conference on machine learning and applications (ICMLA)*, pages 1275–1282. IEEE, 2018. doi: 10.1109/ICMLA.2018.00207.

[133] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2828–2837, 2019. doi: 10.1145/3292500.3330672.

[134] Zijian Niu, Ke Yu, and Xiaofei Wu. Lstm-based vae-gan for time-series anomaly detection. *Sensors*, 20(13):3738, 2020. URL `https://www.mdpi.com/1424-8220/20/13/3738`.

[135] Alexander Geiger, Dongyu Liu, Sarah Alnegheimish, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Tadgan: Time series anomaly detection using generative adversarial networks. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 33–43. IEEE, 2020. doi: 10.1109/BigData50022.2020.9378139.

[136] Haoran Liang, Lei Song, Jianxing Wang, Lili Guo, Xuzhi Li, and Ji Liang. Robust unsupervised anomaly detection via multi-time scale dcgans with forgetting mechanism for industrial multivariate time series. *Neurocomputing*, 423:444–462, 2021. doi: 10.1016/j.neucom.2020.10.084.

[137] Puck de Haan and Sindy Löwe. Contrastive predictive coding for anomaly detection. *arXiv preprint arXiv:2107.07820*, 2021.

[138] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. A dataset to support research in the design of secure water treatment systems. In *International conference on critical information infrastructures security*, pages 88–99. Springer, 2016.

[139] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[140] Aditya P Mathur and Nils Ole Tippenhauer. Swat: A water treatment testbed for research and training on ics security. In *2016 international workshop on cyber-physical systems for smart water networks (CySWater)*, pages 31–36. IEEE, 2016.

[141] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. In *International Conference on Artificial Neural Networks*, pages 703–716. Springer, 2019. doi: 10.1016/j.neucom.2020.10.084. URL `https://www.sciencedirect.com/science/article/pii/S0925231220316970`.

[142] Bin Zhou, Shenghua Liu, Bryan Hooi, Xueqi Cheng, and Jing Ye. Beatgan: Anomalous rhythm detection using adversarially generated time series. In *IJCAI*, pages 4433–4439, 2019.

[143] Lifeng Shen, Zhuocong Li, and James Kwok. Timeseries anomaly detection using temporal hierarchical one-class network. In *Advances in Neural Information Processing Systems*, 2020.

[144] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery*, 29 (3):626–688, 2015.

[145] Lingxiao Zhao and Leman Akoglu. On using classification datasets to evaluate graph outlier detection: Peculiar observations and new insights. *Big Data*, 2021.

[146] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*, pages 5453–5462. PMLR, 2018.

[147] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2018.

[148] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*, 2020. URL `www.graphlearning.io`.

[149] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33:5812–5823, 2020.

[150] Tianle Cai, Shengjie Luo, Keyulu Xu, Di He, Tie-yan Liu, and Liwei Wang. Graphnorm: A principled approach to accelerating graph neural network training. *arXiv preprint arXiv:2009.03294*, 2020.

[151] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*, 2017.

[152] Saurabh Verma and Zhi-Li Zhang. Hunt for the unique, stable, sparse and fast feature learning on graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 87–97, 2017.

[153] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

[154] Marion Neumann, Roman Garnett, Christian Bauckhage, and Kristian Kersting. Propagation kernels: efficient graph kernels from propagated information. *Machine Learning*, 102(2):209–245, 2016.

[155] Tim Schneider, Chen Qiu, Marius Kloft, Decky Aspandi Latif, Steffen Staab, Stephan Mandt, and Maja Rudolph. Detecting anomalies within time series using local neural transformations. *arXiv preprint arXiv:2202.03944*, 2022.

[156] Chen Qiu, Marius Kloft, Stephan Mandt, and Maja Rudolph. Raising the bar in graph-level anomaly detection. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 2196–2203. International Joint Conferences on Artificial Intelligence Organization, 7 2022. Main Track.

[157] Jinsung Yoon, Kihyuk Sohn, Chun-Liang Li, Sercan O Arik, Chen-Yu Lee, and Tomas Pfister. Self-trained one-class classification for unsupervised anomaly detection. *arXiv preprint arXiv:2106.06115*, 2021.

[158] Nico Görnitz, Anne Porbadnigk, Alexander Binder, Claudia Sannelli, Mikio Braun, Klaus-Robert Müller, and Marius Kloft. Learning and evaluation in presence of non-iid label noise. In *Artificial Intelligence and Statistics*, pages 293–302. PMLR, 2014.

[159] Yan Xia, Xudong Cao, Fang Wen, Gang Hua, and Jian Sun. Learning discriminative reconstructions for unsupervised outlier removal. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1511–1519, 2015.

[160] Laura Beggel, Michael Pfeiffer, and Bernd Bischl. Robust anomaly detection in images using adversarial autoencoders. *arXiv preprint arXiv:1901.06355*, 2019.

[161] François-Xavier Aubet, Daniel Zügner, and Jan Gasthaus. Monte carlo em for deep time series anomaly detection. *arXiv preprint arXiv:2112.14436*, 2021.

[162] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer, 1992.

[163] Peter J Huber. Robust statistics. In *International encyclopedia of statistical science*, pages 1248–1251. Springer, 2011.

[164] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? In *International Conference on Learning Representations*, 2018.

[165] Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, volume 3, pages 448–455, 2003.

[166] Tom Shenkar and Lior Wolf. Anomaly detection for tabular data with internal contrastive learning. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=_hszZbt46bT`.

[167] Maxime Alvarez, Jean-Charles Verdier, D'Jeff K Nkashama, Marc Frappier, Pierre-Martin Tardif, and Froduald Kabanza. A revealing large-scale evaluation of unsupervised anomaly detection algorithms. *arXiv preprint arXiv:2204.09825*, 2022.

[168] Guansong Pang, Cheng Yan, Chunhua Shen, Anton van den Hengel, and Xiao Bai. Self-trained deep ordinal regression for end-to-end video anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12173–12182, 2020.

[169] Mahito Sugiyama and Karsten Borgwardt. Rapid distance-based outlier detection via sampling. *Advances in Neural Information Processing Systems*, 26: 467–475, 2013.

[170] Dan Pelleg and Andrew Moore. Active learning for anomaly and rare-category detection. *Advances in neural information processing systems*, 17, 2004.

[171] Md Amran Siddiqui, Alan Fern, Thomas G Dietterich, Ryan Wright, Alec Theriault, and David W Archer. Feedback-guided anomaly discovery via online optimization. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2200–2209, 2018.

[172] Alireza Ghasemi, Hamid R Rabiee, Mohsen Fadaee, Mohammad T Manzuri, and Mohammad H Rohban. Active learning from positive and unlabeled data. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 244–250. IEEE, 2011.

[173] Nico Görnitz, Marius Kloft, Konrad Rieck, and Ulf Brefeld. Toward supervised anomaly detection. *Journal of Artificial Intelligence Research*, 46:235–262, 2013.

[174] Lili Yin, Huangang Wang, and Wenhui Fan. Active learning based support vector data description method for robust novelty detection. *Knowledge-Based Systems*, 153:40–52, 2018.

[175] Shubhomoy Das, Weng-Keen Wong, Thomas Dietterich, Alan Fern, and Andrew Emmott. Incorporating expert feedback into active anomaly discovery. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 853–858. IEEE, 2016.

[176] Shubhomoy Das, Md Rakibul Islam, Nitthilan Kannappan Jayakodi, and Janardhan Rao Doppa. Active anomaly detection via ensembles: Insights, algorithms, and interpretability. *arXiv preprint arXiv:1901.08930*, 2019.

[177] Tiago Pimentel, Marianne Monteiro, Adriano Veloso, and Nivio Ziviani. Deep active learning for anomaly detection. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[178] Jin Ning, Leiting Chen, Chuan Zhou, and Yang Wen. Deep active autoencoders for outlier detection. *Neural Processing Letters*, pages 1–13, 2022.

[179] Xuning Tang, Yihua Shi Astle, and Craig Freeman. Deep anomaly detection with ensemble-based active learning. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 1663–1670. IEEE, 2020.

[180] Stefania Russo, Moritz Lürig, Wenjin Hao, Blake Matthews, and Kris Villez. Active learning for anomaly detection in environmental data. *Environmental Modelling & Software*, 134:104869, 2020.

[181] Guansong Pang, Anton van den Hengel, Chunhua Shen, and Longbing Cao. Toward deep supervised anomaly detection: Reinforcement learning from partially labeled anomaly data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1298–1308, 2021.

[182] Daochen Zha, Kwei-Herng Lai, Mingyang Wan, and Xia Hu. Meta-aad: Active anomaly detection with deep reinforcement learning. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 771–780. IEEE, 2020.

[183] Vincent Barnabé-Lortie, Colin Bellinger, and Nathalie Japkowicz. Active learning for one-class classification. In *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*, pages 390–395. IEEE, 2015.

[184] Holger Trittenbach, Adrian Englhardt, and Klemens Böhm. An overview and a benchmark of active learning for outlier detection with one-class classifiers. *Expert Systems with Applications*, 168:114372, 2021.

[185] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification. *arXiv preprint arXiv:2110.14795*, 2021.

[186] Chen Qiu, Aodong Li, Marius Kloft, Maja Rudolph, and Stephan Mandt. Latent outlier exposure for anomaly detection with contaminated data. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 18153–18167. PMLR, 17–23 Jul 2022.

[187] Naveen Senniappan Karuppusamy and Bo-Yeong Kang. Multimodal system to detect driver fatigue using eeg, gyroscope, and image processing. *IEEE Access*, 8:129645–129667, 2020.

[188] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[189] Chiori Hori, Takaaki Hori, Teng-Yok Lee, Ziming Zhang, Bret Harsham, John R Hershey, Tim K Marks, and Kazuhiko Sumi. Attention-based multimodal fusion for video description. In *Proceedings of the IEEE international conference on computer vision*, pages 4193–4202, 2017.

[190] Xiang Long, Chuang Gan, Gerard Melo, Xiao Liu, Yandong Li, Fu Li, and Shilei Wen. Multimodal keyless attention fusion for video classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[191] Yao Wan, Jingdong Shu, Yulei Sui, Guandong Xu, Zhou Zhao, Jian Wu, and Philip Yu. Multi-modal attention network learning for semantic source code retrieval. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 13–25. IEEE, 2019.

[192] Darshana Priyasad, Tharindu Fernando, Simon Denman, Sridha Sridharan, and Clinton Fookes. Attention driven fusion for multi-modal emotion recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3227–3231. IEEE, 2020.

[193] Yiwei Lu, Frank Yu, Mahesh Kumar Krishna Reddy, and Yang Wang. Few-shot scene-adaptive anomaly detection. In *European Conference on Computer Vision*, pages 125–141. Springer, 2020.

[194] Jhih-Ciang Wu, Ding-Jie Chen, Chiou-Shann Fuh, and Tyng-Luh Liu. Learning unsupervised metaformer for anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4369–4378, 2021.

[195] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[196] Felix Wiewel and Bin Yang. Continual learning for anomaly detection with variational autoencoder. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3837–3841. IEEE, 2019.

[197] Keval Doshi and Yasin Yilmaz. Continual learning for anomaly detection in surveillance videos. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 254–255, 2020.

[198] Keval Doshi and Yasin Yilmaz. Rethinking video anomaly detection-a continual learning approach. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3961–3970, 2022.

[199] Ahmed Frikha, Denis Krompaß, and Volker Tresp. Arcade: A rapid continual anomaly detector. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 10449–10456. IEEE, 2021.

[200] Benjamin Maschler, Thi Thu Huong Pham, and Michael Weyrich. Regularization-based continual learning for anomaly detection in discrete manufacturing. *Procedia CIRP*, 104:452–457, 2021.

[201] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

[202] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33: 18661–18673, 2020.

[203] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.

[204] Yuge Shi, N Siddharth, Philip Torr, and Adam R Kosiorek. Adversarial masking for self-supervised learning. In *International Conference on Machine Learning*, pages 20026–20040. PMLR, 2022.

[205] John Sipple. Interpretable, multidimensional, multimodal anomaly detection with negative sampling for detection of device failure. In *International Conference on Machine Learning*, pages 9016–9025. PMLR, 2020.

[206] Jonas Herskind Sejr and Anna Schneider-Kamp. Explainable outlier detection: What, for whom and why? *Machine Learning with Applications*, 6:100172, 2021.

[207] Vincent Jacob, Fei Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, and Nesime Tatbul. Exathlon: a benchmark for explainable anomaly detection over time series. *Proceedings of the VLDB Endowment*, 14(11):2613–2626, 2021.

[208] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14318–14328, 2022.

[209] Hanqiu Deng and Xingyu Li. Anomaly detection via reverse distillation from one-class embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9737–9746, 2022.

[210] Denis Gudovskiy, Shun Ishizaka, and Kazuki Kozuka. Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 98–107, 2022.

[211] Chin-Chia Tsai, Tsung-Hsuan Wu, and Shang-Hong Lai. Multi-scale patch-based representation learning for image anomaly detection and segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3992–4000, 2022.

[212] Xuefeng Du, Zhaoning Wang, Mu Cai, and Yixuan Li. Vos: Learning what you don't know by virtual outlier synthesis. In *International Conference on Learning Representations*, 2021.

[213] Jonathan G Richens, Ciarán M Lee, and Saurabh Johri. Improving the accuracy of medical diagnosis with causal machine learning. *Nature communications*, 11 (1):1–9, 2020.

[214] Benedek Rozemberczki, Oliver Kiss, and Rik Sarkar. Karate club: an api oriented open-source python framework for unsupervised learning on graphs. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 3125–3132, 2020.

[215] Giannis Siglidis, Giannis Nikolentzos, Stratis Limnios, Christos Giatsidis, Konstantinos Skianis, and Michalis Vazirgiannis. Grakel: A graph kernel library in python. *J. Mach. Learn. Res.*, 21:54–1, 2020.

Chen Qiu

Research Interest

Anomaly Detection; Self-Supervised Learning; Contrastive Learning; Representation Learning; Time Series Modeling

Education

- **Technical University of Kaiserslautern, Germany** *October 2020 - April 2023*
  Ph.D. in Computer Science
- **University of Erlangen-Nuremberg, Germany** *April 2016 - December 2018*
  M.S. in Electrical Engineering
- **Yanshan University, China** *September 2010 - July 2014*
  B.S. in Automation Engineering

Working Experience

- **Bosch Center for Artificial Intelligence, Germany** *May 2019 - October 2022*
  Industrial PhD Candidate: Cutting-edge Research on AI