UNIVERSITY OF CALIFORNIA,
IRVINE

Deep Anomaly Detection and Distribution Shifts

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Aodong Li

Dissertation Committee:
Associate Professor Stephan Mandt, Chair
Professor Padhraic Smyth
Research Professor Maja Rudolph
Professor Erik Sudderth

2024

DEDICATION

To my mother, Mingfang Fu, and my family

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank my advisor, Stephan Mandt, for advising me on my research work during my PhD journey. His influence on me is in all aspects. His advice, feedback, and encouragement taught me research, communication, presentation, and writing skills. More importantly, I learned how to be a responsible researcher. With his help, I published my first top-tier machine-learning conference paper and tasted research success and joy. I still remember during the summers, Stephan used his connections to help me find interesting industrial internships, which allowed me to contrast industrial and academic research. His sufficient funding allows me to put all my energy into research without worrying about intuition and living expenses. Without his help, I couldn't get this far.

My doctoral committee members (Padhraic Smyth, Maja Rudolph, and Erik Sudderth) and Marius Kloft provided me with research advice and guidance at different times. They asked many interesting questions that inspired my research from time to time. They used their experience to help make my ideas concrete and propose writing strategies. I learned a lot.

It has been a pleasure to exchange research ideas with my collaborators, Chen Qiu, Robert Bamler, and Alex Boyd. Talking to them clarified my perplexity and gave me a deeper understanding of the research topics I was working on.

I want to thank Qualcomm, Bosch Center for AI, and Amazon Web Services AI Labs for providing summer internship opportunities to me. I appreciate the guidance from my mentors Hilmi Enes Egilmez (Qualcomm), Maja Rudolph (Bosch), Abishek Sankararaman (Amazon), and Murali Narayanaswamy (Amazon) during my internships.

I would also like to thank Rajesh Ranganath, Dong Wang, and Yao Wang for recommending me for my PhD study. Their recommendation letters opened my research career.

My lab mates and my friends–Ruihan Yang, Yibo Yang, Eliot Wong-Toi, Kushagra Pandey, Yang Meng, Prakhar Srivastava, Justus Will, Tuan Pham Anh, Yunhan Zhao, Chen Yang, Xi Lu, Yicong Huang, Haomiao He, and many others–create a warm and friendly atmosphere that supports my whole five-year stay at UC Irvine.

My family–my wife, Chuyue Wu, my parents, Mingfang Fu and Yuncang Li–and my uncle, Mingxiang Fu, respected my choices and supported my decisions. They encouraged me along the journey. They are my solid foundations.

The material presented in this dissertation was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001120C0021 and the National Science Foundation (NSF) grant numbers 2003237 and 2007719.

VITA

Aodong Li

EDUCATION

Doctor of Philosophy in Computer Science **2024**
University of California, Irvine *Irvine, California*

Master of Science in Computer Science **2019**
New York University *New York, New York*

Master of Engineering in Communication Engineering **2017**
Beijing University of Posts and Telecommunications *Beijing, China*

Bachelor of Engineering in Communication Engineering **2014**
Beijing University of Posts and Telecommunications *Beijing, China*

RESEARCH EXPERIENCE

Graduate Student Researcher **2019–2024**
University of California, Irvine *Irvine, California*

Applied Scientist Intern **2023**
Amazon Web Services AI Labs *Santa Clara, California*

Machine Learning Research Intern (remote) **2022**
Bosch Center for Artificial Intelligence *Pittsburgh, Pennsylvania*

Software Engineering Intern (remote) **2021**
Qualcomm *San Diego, California*

Graduate Research Assistant **2018–2019**
New York University *New York, New York*

TEACHING EXPERIENCE

Teaching Assistant **2019, 2021, 2022**
University of California, Irvine *Irvine, California*

Teaching Assistant **2018, 2019**
New York University *New York, New York*

REFEREED JOURNAL PUBLICATIONS

Brody Foy, Aodong Li, James McClung , Rajesh Ranganath, and John Higgins. Data-Driven Physiologic Thresholds for Iron Deficiency Associated with Hematologic Decline. *American Journal of Hematology*, volume 95(3), pages 302-309. John Wiley & Sons, 2019.

REFEREED CONFERENCE PUBLICATIONS

Aodong Li*, Chen Qiu*, Marius Kloft, Padhraic Smyth, Maja Rudolph†, and Stephan Mandt†. Zero-Shot Anomaly Detection via Batch Normalization. In *Advances in Neural Information Processing Systems*, volume 36, pages 40963–40993. Curran Associates, 2023.

Aodong Li*, Chen Qiu*, Marius Kloft, Padhraic Smyth, Stephan Mandt†, and Maja Rudolph†. Deep Anomaly Detection under Labeling Budget Constraints. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 19882-19910. PRML, 2023.

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, pages 18153-18167. PMLR, 2022.

Aodong Li, Alex Boyd, Padhraic Smyth, and Stephan Mandt. Detecting and Adapting to Irregular Distribution Shifts in Bayesian Online Learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 6816–6828. Curran Associates, 2021.

Aodong Li, Shiyue Zhang, and Dong Wang. Enhanced Neural Machine Translation by Learning from Draft. In *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1583-1587. IEEE, 2017.

* denotes shared first-authorship. † denotes joint supervision.

ABSTRACT OF THE DISSERTATION

Deep Anomaly Detection and Distribution Shifts

By

Aodong Li

Doctor of Philosophy in Computer Science

University of California, Irvine, 2024

Associate Professor Stephan Mandt, Chair

Anomaly detection is important in various applications, from cyber-security, transportation, industry, and finance to healthcare. The anomaly detection problem is to identify anomalies originating from a different data-generating process from normal data. The rare occurrence of anomalies and their unknown causes makes it hard to collect and model them. Thus, anomaly detection methods utilize normal data to build anomaly detectors. In this dissertation, we apply deep anomaly detection methods–methods that apply deep learning techniques–to solve anomaly detection problems. We contribute multiple generic frameworks for various anomaly detection setups.

First, we challenge the common clean training data assumption (free of anomalies) and stress that practical training data is often contaminated with unnoticed anomalies. We propose a novel unsupervised training strategy for training an anomaly detector in the presence of unlabeled anomalies that is compatible with a broad class of models.

Second, selecting informative data points for expert feedback can significantly improve anomaly detection performance. The critical challenges are selecting the most informative samples for expert review and effectively incorporating their feedback to bolster anomaly detection capabilities. To address these challenges, we propose a new data labeling strategy and a new learning framework for active and semi-supervised anomaly detection.

Third, real-world applications may face distribution shifts. We consider the online learning problem where the shifts occur at unknown positions and with unknown intensities. We derive a new Bayesian online inference approach to automatically infer these distribution shifts and adapt the model to the detected changes. This approach applies to both supervised and unsupervised learning settings. We also consider the problem of adapting an anomaly detector to drift in the normal data distribution, especially when no training data is available for the "new normal." This setting is called zero-shot anomaly detection. We propose a simple yet effective method that combines batch normalization and meta-training for zero-shot anomaly detection.

The learning frameworks introduced in this dissertation are model-agnostic and apply to various data types. Extensive experiments demonstrate the efficacy of our proposed approaches.

Chapter 1

Introduction

Anomaly detection, identifying data points that deviate from most collected data, is an important area in machine learning and holds interests and applications in various real-world settings. Examples include scientific discovery [Shapere, 1964], network security [Fernandes et al., 2019], data cleaning [Ilyas and Chu, 2019], financial fraud detection [Hilal et al., 2022], industrial fault detection [Xie et al., 2024, Mokhtari et al., 2021, Bergmann et al., 2019], medical diagnosis [Fernando et al., 2021, Baur et al., 2021], and more.

Anomalies, also referred to as *outliers* or *novelties*[1], are also of significance and interest within the realm of statistics [Kutner et al., 2005, Chapter 3] and data mining [Syarif et al., 2012, Dokas et al., 2002, Agrawal and Agrawal, 2015]. The objective is to detect anomalies within a collection of data. In machine learning, anomaly detection aims to learn to characterize data considered "normal" with associated regions in feature space so that the normal data can be discriminated from data following a different distribution. [2]

Anomalies originate from a different data-generating distribution from normal data. Anomalies occur less frequently, and their causes are unknown, making them difficult or costly to

[1] We differentiate them in detail in Appendix A.

[2] We discuss more details and the connections to out-of-distribution detection in Appendix A.

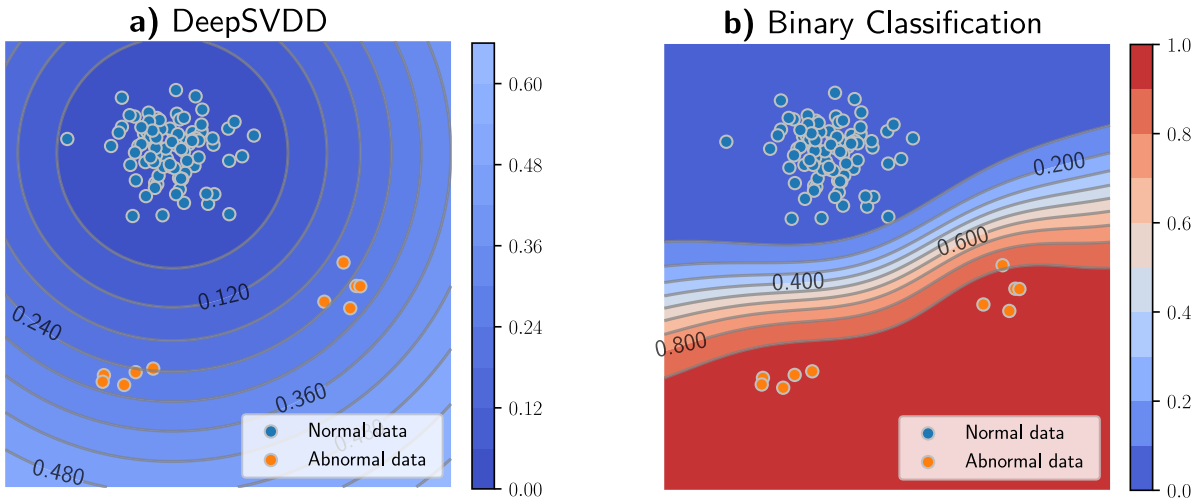


Figure 1.1: Anomaly score contour plots on 2D toy data demonstrate the difference between (**a**) DeepSVDD [Ruff et al., 2018] and (**b**) binary classification. Binary classification (**b**) is problematic for anomaly detection since it cannot detect new anomalies, e.g. in the upper right corner of the plot. DeepSVDD (**a**) relies on an inductive bias that assigns high anomaly scores to regions far from normal data.

acquire, e.g., for discriminative training purposes. Additionally, anomalies may cause severe consequences in some safety-critical applications. These characteristics present challenges for anomaly detection and set it apart from supervised classification. The problem would be easy to solve if one collected sufficiently many representative anomalies. However, due to the difficulties mentioned above, although some anomalous data can be collected in practice, they may be biased and only represent a small portion of the full spectrum of anomalies. Figure 1.1 (b) demonstrates the issues when applying binary classification to anomaly detection.

On the other hand, it is easy to collect normal data. Consequently, anomaly detection primarily leverages normal data for training, aiming to learn a detector to identify previously unseen anomalies at test time. Figure 1.1 (a) presents a concrete example of anomaly detection to contrast Deep Support Vector Data Description (DeepSVDD) [Ruff et al., 2018], an anomaly detection method with binary classification. Unlike discriminative training, anomaly detection methods train exclusively on normal data to learn a parametric anomaly scoring function. The scoring function assigns low scores to normal data while high scores

to abnormal data. Applying a threshold to these scores makes it possible to classify data as normal or anomalous.

Methods for solving anomaly detection tasks span from traditional non-deep learning-based methods [Chandola et al., 2009, Ruff et al., 2021] to deep learning-based methods, which the anomaly detection literature refers to as *deep anomaly detection* [Ruff et al., 2021, Chalapathy and Chawla, 2019, Pang et al., 2021a]. Deep anomaly detection, known for its high expressiveness and the ability to handle complex, high-dimensional data, has emerged as a crucial field spanning various methods. For example, one line of methods aims to learn a neural network that maps all normal data to a pre-defined center in the feature space and maps anomalous data away, referred to as deep one-class classification [Ruff et al., 2018, Liznerski et al., 2020]. Another line of methods is motivated by the advances of deep generative models, thereby modeling the density of normal data Zong et al. [2018], Schlegl et al. [2017], Livernoche et al. [2024]. Self-supervised learning or contrastive learning, able to learn robust and informative representations for downstream tasks [Hendrycks et al., 2019], has been stimulating methods designed for various data types such as images, tables, time series, and so on [Bergman and Hoshen, 2020, Golan and El-Yaniv, 2018, Hendrycks et al., 2019, Qiu et al., 2021, Tack et al., 2020, Shenkar and Wolf, 2021]. Others utilize transfer learning, pre-trained foundation models, or unstructured public datasets to improve existing anomaly detection methods [Deecke et al., 2021, Liznerski et al., 2022, Hendrycks et al., 2018]. In Chapter 2, we will explain the details of some representative methods used in the experiments of this dissertation. While more methods are being developed for various data and anomaly types, this dissertation is not dedicated to novel model architectures. Instead, we propose generic frameworks to enable diverse deep anomaly detection methods to work effectively across various learning environments, e.g., active learning and zero-shot learning.

Anomaly detection tasks vary along with specific scenarios and underlying assumptions. These tasks can be categorized in different aspects. One aspect is the availability of labeled

anomalies in the training data. When the training dataset comprises normal and abnormal data and the anomaly labels are provided, this setup is referred to as *supervised anomaly detection*. This supervised condition is hard to achieve as acquiring the datum-wise labels requires significant human effort. Another common assumption is that a clean, normal dataset clear from anomalies is available at training time. Most of the recent work is developed under this assumption. However, most real-world datasets already contain hidden anomalies, known as corrupted or contaminated data. Contaminated data can occur due to various factors, such as an absence of labeling or a rough screening that only eliminates evident anomalies, often due to limited labor or expertise. Training an anomaly detector with contaminated data is called the *unsupervised contaminated setting*[3]. Consider tumor detection in a medical setting. An automatic detection system will be developed to identify which medical images contain malignant tumors. The training images presented to the system are naturally a mixture of normal and tumor images. Chapter 3 is dedicated to introducing an unsupervised training strategy that enhances deep anomaly detectors under this contaminated training data assumption.

Introducing human-expert interaction during training can enhance performance, especially if informative data points are chosen for expert labeling. In the same tumor detection example, this human-involvement setup suggests that the anomaly labels of some images are available through a doctor. The critical challenges are how to select the most informative samples for expert review and effectively incorporate their feedback to bolster anomaly detection capabilities[4]. This approach, blending active querying and semi-supervised techniques, is often termed active and semi-supervised deep anomaly detection. In Chapter 4, we present a framework for deep anomaly detection under limited expert labeling budgets.[5]

[3]Some literature refers to this setting as "unsupervised anomaly detection."

[4]with an assumption that labeled anomalies are informative and reflect some anomalous modes.

[5]In the literature, some works refer to the clean training data setup as "semi-supervised anomaly detection." We need clarification on these names, e.g., semi-supervised anomaly detection is a larger concept than merely ensuring a clean training dataset. Therefore, we instead adopt our naming standards, i.e., using "clean training dataset" or "semi-supervised anomaly detection" based on context.

Additionally, the tumor detector may face distribution shifts. The medical images may have been recorded with new imaging technologies and formats, or the patients are from a different demographic the anomaly detector has not seen. The notion of what constitutes normal data can shift, and anomalies might evolve in response to detection efforts adversarially (e.g., in cybersecurity). These data shifts challenge adapting the detection model to new data distributions. To tackle this problem, Chapter 6 introduces a generic adaptation framework with Bayesian online learning in a data-streaming environment. The framework applies to both supervised and unsupervised learning.

Finally, acquiring sufficient training data from new distributions proves formidable, and fast adaptation is preferred. Various solutions within the realms of few-shot and zero-shot learning have been proposed. Chapter 5 contributes a new lightweight zero-shot anomaly detection framework compatible with various deep anomaly detection methods and multiple data types.

1.1 Dissertation Organization

As follows, Chapter 2 introduces the problem setup and representative deep anomaly detection methods, setting the stage for the rest of this dissertation. In Chapter 3, we introduce a novel framework called Latent Outlier Exposure that improves the performance of deep anomaly detection methods in the contaminated training dataset setting. Chapter 4 develops an effective active learning strategy and a novel semi-supervised anomaly detection objective for deep anomaly detection under labeling budget constraints. Chapter 5 introduces a lightweight zero-shot anomaly detection framework with batch normalization. Chapter 6 provides a general adaptation framework compatible with Bayesian online learning. Finally, Chapter 7 concludes this dissertation.

Chapter 2

Background

In this chapter, we start by setting up the notation and defining the anomaly detection task, followed by an overview of several widely used deep anomaly detection techniques. Additional sections set up the stage for subsequent chapters. Section 2.3 provides semi-supervised anomaly detection background for Chapters 3 and 4. Section 2.4.1 introduces Bayesian online learning and variational continual learning for Chapter 6. Section 2.4.2 gives an overview of few-shot and zero-shot anomaly detection for Chapter 5.

2.1 Notation and Problem Statement

In the following, we borrow notations from Ruff et al. [2021]. Assume the probability density functions of normal and abnormal data distributions exist, denoted by $p_n(\mathbf{x})$ and $p_a(\mathbf{x})$, respectively. Let $\mathcal{X} \in \mathbb{R}^D$ denote the data space of interest. Consider a dataset $\{\mathbf{x}_i \in \mathcal{X}\}_{i=1}^N$ where data points $\mathbf{x}_i$ are i.i.d. samples from a mixture distribution

$$p(\mathbf{x}) = (1-\alpha)p_n(\mathbf{x}) + \alpha p_a(\mathbf{x}) \tag{2.1}$$

where $p_n(\mathbf{x})$ is the normal data distribution and $p_a(\mathbf{x})$ corresponds to the abnormal data.

We assume the anomalous data is non-dominant in the mixture, i.e., the contamination ratio is small $0 \leq \alpha < 0.5$. In Section 2.2, we make the most common assumption that the dataset is clean and only contains normal data, i.e., $\alpha = 0$. We will relax the clean training data assumption in subsequent sections and chapters and allow $\alpha > 0$. Each data point is possibly labeled normal or anomalous, as indicated by the binary anomaly label $y_i := y(\mathbf{x}_i) \in \{0 := \text{“normal”}, 1 := \text{“abnormal”}\}$.

The anomaly detection problem is identifying the low-probability region under the normal data distribution. The region corresponds to $\mathcal{A} := \{\mathbf{x} \in \mathcal{X} | p_n(\mathbf{x}) \leq \gamma\}$. $\gamma \geq 0$ is a low-density region threshold. Adjusting the threshold trades off the false positive against the false negative rates, which can be customized based on the task preference.

While the low-density region can be unbounded, an underlying assumption in anomaly detection is that the normal data concentrates in a high-density region. The high-density region corresponds to a bounded density level set $\{\mathbf{x} \in \mathcal{X} | p_n(\mathbf{x}) > \gamma\}$. This assumption is known as the Concentration Assumption [Ruff et al., 2021]. More concretely, we consider learning desired density level sets of normal data. Two options are available. One is to learn a density estimator based on the given dataset. One can model the normal data density $p_n(\mathbf{x})$ with a parametric density function $p(\mathbf{x}; \theta)$. Deep generative models[1] like variational auto-encoders (VAEs)[2] [Kingma and Welling, 2014, Rezende et al., 2014] and flow-based models [Kingma and Dhariwal, 2018], are utilized for density estimation and, in turn, for anomaly detection by applying a threshold to the log-likelihood of the learned model [Nalisnick et al., 2019]. However, generative models often perform poorly for complex data, e.g., CIFAR-10, due to their tendency to model low-level statistics, overlook high-level semantics, and assign higher

[1]Generative adversarial networks (GANs) [Goodfellow et al., 2014] are not included as their likelihoods are hard to evaluate [Bond-Taylor et al., 2021]. That said, the discriminator of GANs can be used for anomaly detection [Schlegl et al., 2017].

[2]VAE models a lower bound of $p_n(\mathbf{x})$. The lower bound often serves as a surrogate of $p_n(\mathbf{x})$.

likelihoods to simpler datasets [Nalisnick et al., 2019].

Another line of research learns to output a transformation of the data density function $T(p_n(\mathbf{x}))$, in the form of a parametric *anomaly scoring* function $S(\mathbf{x};\theta) \in \mathbb{R}$ such that the anomaly scores preserve the ranking between normal and anomalous data, that is, $S(\mathbf{x}_a;\theta) > S(\mathbf{x}_n;\theta)$ if[3] $y(\mathbf{x}_n) = 0$ and $y(\mathbf{x}_a) = 1$. $T(p_n(\mathbf{x}))$ or $S(\mathbf{x};\theta)$ is not necessarily a probability density function. The non-probabilistic relaxation eases the problem as learning the anomaly score function $S(\mathbf{x};\theta)$ does not require the normalization constant to be learned[4]. Once trained, the output anomaly scores can be applied with a threshold to identify anomalies. This chapter focuses on designing and learning the parametric anomaly score functions.

Evaluation Metric. Multiple metrics can be used to evaluate the performance of a learned anomaly detector, e.g., AUROC, F1 score, and AUPR. We mainly rely on AUROC to evaluate an anomaly detector $S(\mathbf{x};\theta)$ as AUROC directly evaluates the bipartite ranking quality [Mohri et al., 2018, 10.5.2][Cortes and Mohri, 2003]. It estimates the probability of ranking abnormal data higher than normal data in terms of their scores, i.e., $\mathbb{E}_{\mathbf{x}_a\sim p_a(\mathbf{x}),\mathbf{x}_n\sim p_n(\mathbf{x})}[\mathbb{1}(S(\mathbf{x}_a;\theta) > S(\mathbf{x}_n;\theta))]$.

2.2 Deep Anomaly Detection

Deep anomaly detection exploits the feature representation ability of deep neural networks to detect anomalies in complex and high-dimensional data such as images or videos. This section will review some deep anomaly detection methods used in this thesis's experiments. We refer readers for more deep anomaly detection methods to review articles [Chalapathy

[3]Here, we assume the uniqueness of the anomaly label.

[4]Consider an energy-based model $p(\mathbf{x};\theta) = \exp(-S(\mathbf{x};\theta))/Z(\theta)$ with the normalization constant $Z(\theta) = \int \exp(-S(\mathbf{x};\theta))d\mathbf{x}$. The anomaly score function $S(\mathbf{x};\theta)$ is an energy function, and $p(\mathbf{x};\theta)$ is a probability density. Since we only optimize $S(\mathbf{x};\theta)$, we do not evaluate the normalization constant during training, which makes the learning problem easier than fitting a density model.

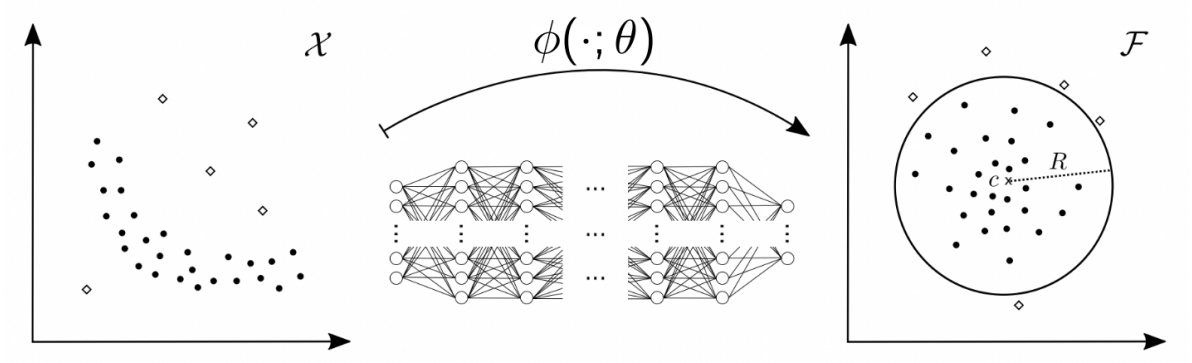


Figure 2.1: DeepSVDD learns a deep neural network-based transformation $\phi(\cdot;\theta)$ that maps a data point in data space $\mathcal{X}$ to a corresponding point in feature space $\mathcal{F}$. DeepSVDD operates such that in the feature space, all normal data (black dots) are mapped to a tight neighborhood of the center $\mathbf{c}$. The size of the neighborhood is measured by its radius R. Minimizing R is equivalent to minimizing each normal data's distance to the center $\mathbf{c}$, which is the objective function Equation (2.2). Image adapted from Ruff et al. [2018].

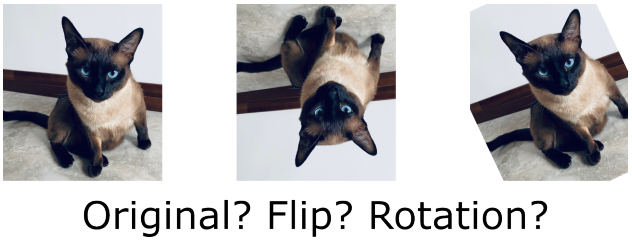


Figure 2.2: Self-supervised learning for images predicts various image augmentations. For example, an image can be flipped or rotated and then used as input to a classifier to decide if a type of augmentation is applied. The learned classifier can be used for anomaly detection.

and Chawla, 2019, Pang et al., 2021a, Ruff et al., 2021].

Common deep anomaly detection methods rely on one-class classification or self-supervised learning techniques to learn an anomaly score function that preserves the ranking between normal and abnormal data. The anomaly score is a parametric function $S(\mathbf{x};\theta)$ where the learnable parameters θ are neural network weights. We introduce some deep anomaly detection methods in the clean training data setup in the following. These methods differ in the self-supervised learning objectives.

Deep Support Vector Data Description (DeepSVDD) [Ruff et al., 2018] is a deep learning-enabled version of one-class support vector machine [Schölkopf et al., 2001, Tax

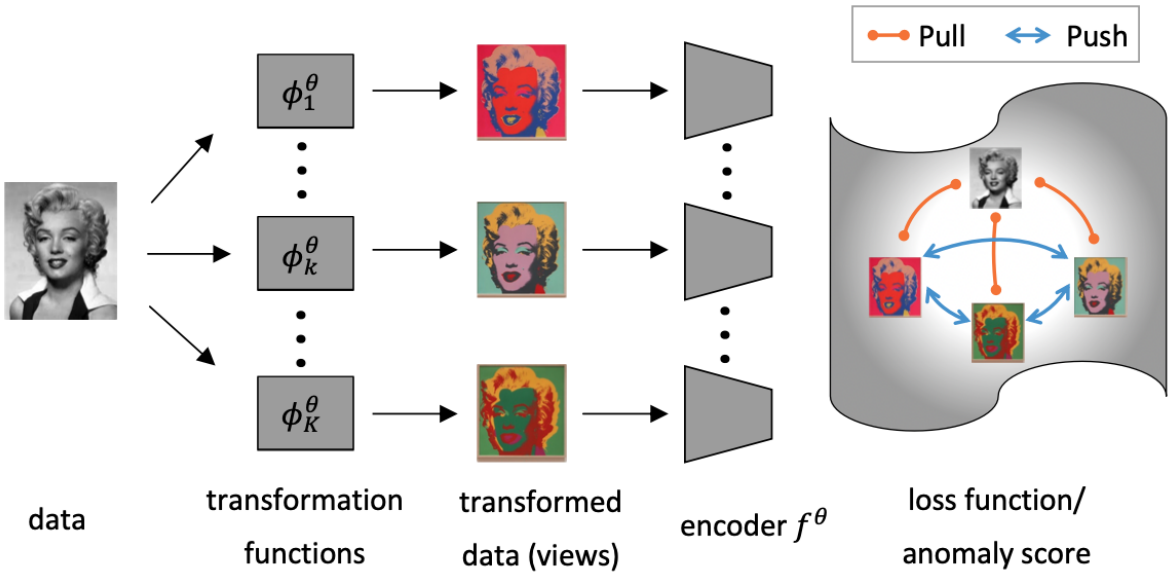


Figure 2.3: NTL applies neural networks to learn diverse views (neural transformations) of the original data and performs contrastive learning. The transformed and original data are then encoded into a feature space such that each transformed data is supposed to be close to the original data and far from the other. The benefit of using learnable transformations is that the method applies to various data types such as time series and tabular data that lack manually designed augmentations, in contrast with image data whose augmentations are designed by human experts. Images are adapted from Qiu et al. [2021].

and Duin, 2004b]. The idea of DeepSVDD is to learn a parametric feature representation $\phi(\mathbf{x};\theta) \in \mathbb{R}^k$ such that, in the representation space, all normal training data is mapped to either a pre-defined or learned center $\mathbf{c} \in \mathbb{R}^k$. Since the model only saw normal data during training, anomalies will not be mapped to the same vicinity by default. Figure 2.1 illustrates this idea. In the implementation, the feature representation mapping $\phi(\mathbf{x};\theta)$ is an instantiation of a multi-layer neural network. One can learn feature mapping by minimizing the Euclidean distance between the mapped representation vector of each normal data point and the center

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}^{\theta}_{\text{SVDD}}(\mathbf{x}_i)$$
$$\text{with } \mathcal{L}^{\theta}_{\text{SVDD}}(\mathbf{x}) = \|\phi(\mathbf{x}_i;\theta) - \mathbf{c}\|_2^2. \tag{2.2}$$

Once trained, the same Euclidean distance is also used as the anomaly score to identify anomalies, i.e., $S(\mathbf{x};\theta) = \mathcal{L}^{\theta}_{\text{SVDD}}(\mathbf{x})$. Only normal data is employed to minimize the anomaly score during training. The anomalies will exhibit higher scores than normal data. For more details on parameter initialization and selection of the center $\mathbf{c}$, we refer readers to Ruff et al. [2018].

Multi-Head Rotation Net (MHRot) [Hendrycks et al., 2019] amounts to learning a multi-head classifier to predict whether an image is augmented by handcrafted transformations such as rotations, horizontal shifts, and vertical shifts. Figure 2.2 provides some common examples of image augmentations used in self-supervised learning. Suppose M different transformations are available. Each image can be applied with multiple transformations, giving various augmentations due to transformation compositions. Since each transformation can be added or not independently, the number of transformation compositions is $K = 2^M$. Denote these compositions by $\{T_k\}_{k=1}^{K}$. The multi-head classifier has M prediction heads with parameters θ_m, each representing $p(t_{k,m}|T_k(\mathbf{x});\theta_m)$, where $t_{k,m} \in \{0,1\}$ indicates whether or not a transformation m is used[5] in T_k. Aiming to predict the correct transformations for normal samples, we minimize the negative log-likelihoods of the ground truth label $t_{k,m}$ for each transformation m and for each transformation composition T_k, resulting in the loss function for each data point $\mathbf{x}$

$$\mathcal{L}^{\theta}_{\text{MHRot}}(\mathbf{x}) := -\sum_{k=1}^{K}\sum_{m=1}^{M} \log p(t_{k,m}|T_k(\mathbf{x});\theta_m). \tag{2.3}$$

The loss function on the training dataset is $\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}^{\theta}_{\text{MHRot}}(\mathbf{x}_i)$. At test time, the anomaly score is the same as the loss function $S(\mathbf{x};\theta) = \mathcal{L}^{\theta}_{\text{MHRot}}(\mathbf{x})$, or the entropy[6] of the model predictions $S(\mathbf{x};\theta) = \sum_{k=1}^{K}\sum_{m=1}^{M} H(p(\cdot|T_k(\mathbf{x});\theta_m))$. Since abnormal data is not observed during training, the anomaly score is expected to be higher for anomalous data

[5] In practice, multi-class classification is also possible for each head. For example, various rotation angles can be applied to augment an image, and then the head predicts which angle is applied.

[6] The entropy of a discrete probability distribution is $H(p(x)) = -\sum_x p(x)\log p(x)$.

than normal data at test time.

Neural Transformation Learning (NTL). Rather than using hand-crafted transformations, NTL [Qiu et al., 2021] learns K neural transformations $\{T_{\theta,1}, ..., T_{\theta,K}\}$ and an encoder f_θ from data and uses the learned transformations to detect anomalies. Each neural transformation generates a view $\mathbf{x}_k = T_{\theta,k}(\mathbf{x})$ of sample $\mathbf{x}$. For normal samples, NTL encourages each transformation to be similar to the original sample and to be dissimilar from other transformations. To achieve this objective, NTL maximizes the normalized outputs $z_k(\mathbf{x};\theta) = h(\mathbf{x}_k, \mathbf{x};\theta) \big/ \big(h(\mathbf{x}_k, \mathbf{x};\theta) + \sum_{l\neq k} h(\mathbf{x}_k, \mathbf{x}_l;\theta)\big)$ for each view where $h(\mathbf{a}, \mathbf{b};\theta) = \exp(\cos(f_\theta(\mathbf{a}), f_\theta(\mathbf{b}))/\tau)$ measures the similarity of two views[7],

$$\mathcal{L}^\theta_{\text{NTL}}(\mathbf{x}) := -\sum_{k=1}^{K} \log z_k(\mathbf{x};\theta).$$

Similar to MHRot and DeepSVDD, the overall loss function is an average of losses incurred by each training data point. The anomaly score is the same as the loss function used during training $S(\mathbf{x};\theta) = \mathcal{L}^\theta_{\text{NTL}}(\mathbf{x})$. Figure 2.3 provides an overview of NTL.

2.3 Semi-Supervised Anomaly Detection

In semi-supervised anomaly detection, most of the training data is unlabeled, while a small portion is annotated with anomaly labels. A semi-supervised learning approach is effective if the labeled anomalies in the training dataset are informative, meaning they reflect some modes of abnormality. Incorporating these identified anomalous modes into deep anomaly detection models can enhance its performance.

Denote the index set of labeled and unlabeled data by $\mathcal{Q}$ and $\mathcal{U}$, respectively. Ruff et al. [2019]

[7] where τ is the temperature and $\cos(\mathbf{a}, \mathbf{b}) := \mathbf{a}^\top \mathbf{b}/\|\mathbf{a}\|\|\mathbf{b}\|$

proposed Deep Semi-supervised Anomaly Detection (Deep SAD) that extends DeepSVDD to utilize labeled and unlabeled data during training. The objective is

$$\begin{aligned}\mathcal{L}(\theta) =& \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \underbrace{\mathcal{L}^{\theta}_{\text{SVDD}}(\mathbf{x}_i)}_{\text{unsupervised loss}} \\ &+ \frac{\lambda}{|\mathcal{Q}|} \sum_{j \in \mathcal{Q}} \underbrace{(1-y_j)\mathcal{L}^{\theta}_{\text{SVDD}}(\mathbf{x}_j) + y_j \left(\mathcal{L}^{\theta}_{\text{SVDD}}(\mathbf{x}_j)\right)^{-1}}_{\text{supervised loss}}. \end{aligned} \tag{2.4}$$

The supervised loss concentrates the labeled normal data around the center in the feature space and maps the known abnormal data far away from the center by minimizing the inverse distance. When no labels are available, i.e., $\mathcal{Q}$ is empty, Deep SAD regresses to DeepSVDD (Equation (2.2)). λ, a trade-off hyperparameter, controls how much the learned model focuses on the labeled data points.

Outlier Exposure (OE) [Hendrycks et al., 2018] is another semi-supervised method that uses public image data as auxiliary outliers to improve the performance of anomaly detection methods. The difference from Deep SAD lies in the formation of unlabeled and labeled sets $\mathcal{U}$ and $\mathcal{Q}$. While Deep SAD assumes all data is from a mixture of normal and abnormal data (Equation (2.1)), OE assumes a clean normal dataset and synthesizes anomalies randomly sampled from large unrelated public datasets.

The first unsupervised loss in Equation (2.4) suggests Deep SAD regards all unlabeled data as normal and leaves unlabeled anomalous data fitted by the model, leading to the inability to detect similar anomalies at test time. In Chapters 3 and 4, we introduce new training procedures and objective functions that utilize the information of unlabeled anomalies in the training dataset to improve a deep anomaly detector.

Another assumption Deep SAD makes is that a subset of labeled data exists. This assumption raises the question of whether or not choosing a more informative subset for labeling could boost the performance of semi-supervised anomaly detection. The concept of actively

selecting data for labeling based on the model's current training state introduces us to *active anomaly detection*. A commonly adopted approach is the positive querying strategy, which prefers selecting data points exhibiting higher anomaly scores, as highlighted in the literature [Trittenbach et al., 2021]. Once the queried data points are labeled, a semi-supervised training objective is employed to refine the anomaly detection capabilities further. In Chapter 4, we derive theoretical conditions under which anomaly scores generalize from labeled queries to unlabeled data. Motivated by these results, we propose a data labeling strategy with optimal data coverage under labeling budget constraints.

2.4 Adapting to Shifts in Data Distributions

Real-world data distributions may change over time and space. Adaptation is, for instance, required for security applications when malicious attacks evolve to avoid firewalls. Patients may present demographic disparities in medical domains, and medical images with different imaging technologies vary in resolution, lightness, and preprocessing procedures. In the following, we set the grounds for Chapters 5 and 6: one online learning setup where data from a distribution after shift is available for updating the model and another fast-adaptation setting without updating the model parameters.

2.4.1 Variational Continual Learning

In this part, we consider the problem of online learning in the presence of distribution shifts of unknown strengths (for example, as measured by Kullback-Leibler distance, Wasserstein distance, or total variation distance). We stress that multiple distribution shifts may occur over time. A Bayesian approach to this problem is Variational Continual Learning (VCL) [Nguyen et al., 2018b], a recursive method applicable to supervised and unsupervised

learning setups. VCL relies on the capacity of deep neural networks and the regularization provided by Bayesian online learning to remember different tasks to achieve adaptations to distribution shifts.

Variational Inference. Before presenting the recursive methods to the Bayesian online learning problem, we introduce *variational inference* (VI) [Blei et al., 2017, Zhang et al., 2018], an approximate inference technique to Bayesian inference when the exact posterior is intractable.

Consider observing dataset $\mathcal{D}$ and assuming a data-generating model $p(\mathcal{D}|\theta)$ for $\mathcal{D}$. Adopting a Bayesian treatment, we put a prior distribution $p(\theta)$ over the model parameters and infer their posterior $p(\theta|\mathcal{D})$ conditioned on observing $\mathcal{D}$.

Most interesting models lack a tractable posterior for the parameters. We need to estimate their posterior. One approach is variational inference. One first specifies a variational family $\mathcal{Q}$ that is a class of probability distributions $q(\theta;\lambda)$ with parameters λ. $q(\theta;\lambda)$ and λ are called variational distributions and variational parameters respectively[8]. For instance, a frequent choice of the variational family is Gaussian distributions $\mathcal{N}(\theta;\mu,\Sigma)$ with mean μ and covariance matrix Σ. The variational parameters of the Gaussian family are $\lambda := \{\mu,\Sigma\}$. We aim to search a particular distribution $q \in \mathcal{Q}$ among the variational family to approximate the exact posterior $p(\theta|\mathcal{D})$ by optimizing an objective function (derived below). Selecting $q(\theta;\lambda)$ corresponds to finding an instance of the variational parameter λ that governs the distribution shape.

Next, we derive the objective function used to select the variational distribution q that approximates the posterior $p(\theta|\mathcal{D})$. We derive a lower bound of the marginal probability of

[8]We abuse the notation $\mathcal{Q}$, which denoted the labeled index set in active anomaly detection (Section 2.3). Their meanings are clear from the context.

observations

$$\log p(\mathcal{D}) = \log \int p(\mathcal{D}|\theta)p(\theta)d\theta \tag{2.5}$$

$$= \log \int \frac{p(\mathcal{D}|\theta)p(\theta)}{q(\theta;\lambda)} q(\theta;\lambda)d\theta \tag{2.6}$$

$$\geq \int q(\theta;\lambda) \log \frac{p(\mathcal{D}|\theta)p(\theta)}{q(\theta;\lambda)} d\theta \qquad \text{(By Jensen's inequality)} \tag{2.7}$$

$$= \mathbb{E}_{q(\theta;\lambda)} \left[\log p(\mathcal{D}|\theta)p(\theta) - \log q(\theta;\lambda)\right] := \mathcal{L}(\lambda) \tag{2.8}$$

Equation (2.8) is called the *Evidence Lower BOund*, or ELBO, denoted by $\mathcal{L}(\lambda)$. We can maximize the ELBO with respect to λ to get an optimal variational distribution $q(\theta;\lambda^*)$ that maximizes $\mathcal{L}(\lambda)$ as well as tighten the lower bound of the marginal probability of the observations.

This optimal variational distribution $q(\theta;\lambda^*)$ approximates the posterior $p(\theta|\mathcal{D})$ of interest, which becomes clear if we re-write the ELBO

$$\mathcal{L}(\lambda) := \mathbb{E}_{q(\theta;\lambda)} \left[\log p(\mathcal{D}|\theta)p(\theta) - \log q(\theta;\lambda) + \log p(\mathcal{D}) - \log p(\mathcal{D})\right] \tag{2.9}$$

$$= \mathbb{E}_{q(\theta;\lambda)} \left[\log p(\theta|\mathcal{D}) - \log q(\theta;\lambda)\right] + \log p(\mathcal{D}) \tag{2.10}$$

$$= -D_{\mathrm{KL}}(q(\theta;\lambda)|p(\theta|\mathcal{D})) + \log p(\mathcal{D}) \tag{2.11}$$

where $D_{\mathrm{KL}}(q|p) = \mathbb{E}_q[\log q - \log p]$ is the Kullback–Leibler divergence which satisfies $D_{\mathrm{KL}}(q|p) \geq 0$ and $D_{\mathrm{KL}}(q|p) = 0$ only if $q = p$. It is clear from the last equation that maximizing the ELBO is equivalent to minimizing the KL divergence between the variational distribution $q(\theta;\lambda)$ and the true posterior probability $p(\theta|\mathcal{D})$.

Bayesian Online Learning. Given a sequence of datasets $\{\mathcal{D}_t\}_{t=1}^T$, which are i.i.d. sampled from a stationary data distribution. Dataset $\mathcal{D}_t$ arrives at time t, and the model parameters are updated upon the data arrival. The datasets can be either for discriminative

learning with labels $\mathcal{D}_t = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N_t}$ or for unsupervised learning $\mathcal{D}_t = \{\mathbf{x}_i\}_{i=1}^{N_t}$. The likelihood model $p(\mathcal{D}_t|\theta)$ reflects the differences: assuming i.i.d. samples, discriminative learning employs conditional distribution $p(\mathcal{D}_t|\theta) = \prod_i p(y_i|\mathbf{x}_i, \theta)$ and unsupervised learning uses $p(\mathcal{D}_t|\theta) = \prod_i p(\mathbf{x}_i|\theta)$. Bayesian online learning places a prior distribution $p(\theta)$ over neural network weights and recursively incorporates each dataset by updating the posterior of θ. In particular, it applies the posterior at the previous time step $t-1$ as the prior distribution for the current time step t,

$$p(\theta|\mathcal{D}_{1:t}) \propto p(\mathcal{D}_t|\theta)p(\theta|\mathcal{D}_{1:t-1}), \quad t = 1, \ldots, T. \tag{2.12}$$

Variational Continual Learning [Nguyen et al., 2018b]. VCL instead assumes each dataset can be displaced from the previous one due to a time-varying data-generating process. Adapting to each dataset relies on the capacity of deep neural networks and the regularization provided by Bayesian online learning to remember different datasets. The posterior distribution is intractable in most cases. Thus VCL applies *variational inference* (VI) [Blei et al., 2017, Zhang et al., 2018] to approximate the true posterior for each dataset $\mathcal{D}_t$ at time step t. That is, it iteratively finds the optimal approximate posterior $q_t(\theta)$ for dataset $\mathcal{D}_t$ among a set of variational distributions $\mathcal{Q}$

$$q_t(\theta) = \arg\min_{q \in \mathcal{Q}} D_{\mathrm{KL}} \left[q(\theta) \middle| \frac{q_{t-1}(\theta)p(\mathcal{D}_t|\theta)}{Z_t} \right], \quad t = 1, \ldots, T. \tag{2.13}$$

Z_t is an intractable normalizing constant of $q_{t-1}(\theta)p(\mathcal{D}_t|\theta)$ which does not affect the minimizing of the KL divergence with respect to q. VCL sets $q_0(\theta) := p(\theta)$ for $t = 1$. The KL divergence in Equation (2.13) is also called negative *evidence lower bound* (ELBO). Gaussian distribution is a common choice for $\mathcal{Q}$. Optimizing Equation (2.13) is often intractable and requires Monte Carlo VI [Blundell et al., 2015].

The fact that VCL assumes a non-stationary data-generating process but uses the posterior

at the previous time step as the new prior at the current time step suggests an inconsistency between the i.i.d. modeling assumption and the non-stationary data-generating process assumption. The i.i.d. modeling assumption leads to slower adaptation over time. In Chapter 6, we propose a Bayesian adaptation framework for supervised or unsupervised learning to adjust a model to irregular distribution shifts. The adaptation is achieved by automatically retaining and discarding previous information as necessary.

2.4.2 Few-Shot and Zero-Shot Anomaly Detection

We now refocus on the specific anomaly detection problem under distribution shifts. We further assume the model lacks access to substantial training data from new distributions, thus precluding re-training of the model. This assumption is common in practice. For example, a tumor detection system may encounter medical images produced by a new imaging technology.

One line of research for fast adaptation is to exploit existing few-shot learning or meta-learning techniques for anomaly detection. These techniques enable the model to adapt using a small support set from a new distribution at test time. For instance, model-agnostic meta-learning [Finn et al., 2017] has been applied for one-class classification [Frikha et al., 2021], GANs [Lu et al., 2020b], and autoencoders [Wu et al., 2021b] to detect anomalies in a changing environment.

Unlike few-shot anomaly detection, zero-shot anomaly detection requires no support set to adapt a model. Most zero-shot anomaly detection methods rely on the inherent zero-shot learning capabilities of pre-trained foundational models on images, as demonstrated in studies [Radford et al., 2021, Yu et al., 2022, Jia et al., 2021, Yuan et al., 2021]. An instance is the Contrastive Language–Image Pre-training (CLIP) by Radford et al. [2021], which learns visual representations by leveraging a vast collection of open-source images paired

with natural-language text descriptions. The resulting network projects visual images and language descriptions into a shared feature space. The pre-trained model can provide meaningful representations for downstream tasks such as image classification and anomaly detection. In the context of zero-shot anomaly detection, as explored by Liznerski et al. [2022], CLIP employs a unique approach by comparing test images to a pair of natural language descriptions for normal and abnormal data: $\{l_n =$ "A photo of {NORMAL_CLASS}", $l_a =$ "A photo of something"$\}$. The anomaly score of a test image $\mathbf{x}$ is the relative distance between $\mathbf{x}$ to l_n and $\mathbf{x}$ to l_a in the feature space,

$$S(\mathbf{x}; \theta_{\text{CLIP}}) = \frac{\exp(\langle f_x(\mathbf{x}), f_l(l_a) \rangle)}{\sum_{c \in \{l_n, l_a\}} \exp(\langle f_x(\mathbf{x}), f_l(c) \rangle)}, \tag{2.14}$$

where f_x and f_l are the CLIP image and description feature extractors and $\langle \cdot, \cdot \rangle$ is the inner product. We call this method CLIP-AD.

CLIP-AD requires a relevant language description for the image; an assumption may only be practical for some image datasets. For example, in Omniglot Lake et al. [2015], describing the written alphabet in natural language can be challenging. Moreover, CLIP-AD is unsuitable for other data types like tabular or time-series data. Additionally, its ability to adapt to new distributions beyond its initial training data distribution is limited. To address these issues, Chapter 5 introduces a novel lightweight zero-shot anomaly detection method that circumvents the limitations associated with foundation models.

Chapter 3

Latent Outlier Exposure for Anomaly Detection with Contaminated Training Data

This chapter is based on a published paper at ICML 2022: *Latent Outlier Exposure for Anomaly Detection with Contaminated Data* by Chen Qiu*, Aodong Li*, Marius Kloft, Maja Rudolph, Stephan Mandt [Qiu et al., 2022b]

3.1 Introduction

From industrial fault detection to medical image analysis or financial fraud prevention: Anomaly detection—the task of automatically identifying anomalous data instances without being explicitly taught how anomalies may look like—is critical in industrial and technological applications.

The common approach in deep anomaly detection is to first train a neural network on a large

dataset of "normal" samples minimizing some loss function (such as a deep one-class classifier (DeepSVDD) [Ruff et al., 2018]) and then construct an anomaly score from the output of the neural network (typically based on the training loss). Anomalies are then identified as data points with larger-than-usual anomaly scores and obtained by thresholding the score at particular values.

A standard assumption in this approach is that clean training data are available to teach the model what "normal" samples look like [Ruff et al., 2021]. 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 dataset of medical images may already contain cancer images, or datasets of financial transactions could already contain unnoticed fraudulent activity. Naively training an unsupervised anomaly detector on such data may suffer from degraded performance.

In this chapter, we introduce a new unsupervised approach to training anomaly detectors on a corrupted dataset. Our approach uses a combination of two coupled losses to extract learning signals from both normal and anomalous data. We stress that these losses do not necessarily have a probabilistic interpretation; rather, many recently proposed self-supervised auxiliary losses can be used [Ruff et al., 2018, Hendrycks et al., 2019, Qiu et al., 2021, Shenkar and Wolf, 2022]. 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 a threshold on only one of the two loss functions to identify anomalies in constant time.

Our approach can be applied to a variety of anomaly detection loss functions and data types, as we demonstrate on 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. Compared to established baselines that either ignore the anomalies or try to iteratively remove them [Yoon et al., 2021], our approach yields significant performance improvements in all cases.

The chapter is structured as follows. In Section 3.2, we discuss related work. In Section 3.3, we introduce our main algorithm, including the involved losses and optimization procedure. Finally, in Section 3.4, we discuss experiments on both image and tabular data and discuss our findings in Section 3.5 [1].

3.2 Related Work

We divide our related work into methods for deep anomaly detection, learning on incomplete or contaminated data, and training anomaly detectors on contaminated data.

Deep anomaly detection. Deep learning has played an important role in recent advances in anomaly detection. For example, Ruff et al. [2018] have improved the anomaly detection accuracy of one-class classification [Schölkopf et al., 2001] by combining it with a deep feature extractor, both in the unsupervised and the semi-supervised setting [Ruff et al., 2019]. An alternative strategy to combine deep learning with one-class approaches is to train a one-class SVM on pretrained self-supervised features [Sohn et al., 2020b]. Indeed, self-supervised learning has influenced deep anomaly detection in a number of ways: The self-supervised criterion for training a deep feature extractor can be used directly to score anomalies [Golan and El-Yaniv, 2018, Bergman and Hoshen, 2020]. Using a MHRot, Hendrycks et al. [2019] improve self-supervised anomaly detection by solving multiple classification tasks. For general data types beyond images, NTL [Qiu et al., 2021, 2022a] learns the transformations for

[1] Code is available at `https://github.com/aodongli/Latent-Outlier-Exposure` and `https://github.com/boschresearch/LatentOE-AD.git`

the self-supervision task and achieves solid detection accuracy. Schneider et al. [2022] combine NTL with representation learning for detecting anomalies within time series. On tabular data, anomaly detection with internal contrastive learning (ICL) [Shenkar and Wolf, 2022] learns feature relations as a self-supervised learning task. Other classes of deep anomaly detection includes autoencoder variants [Principi et al., 2017, Zhou and Paffenroth, 2017, Chen and Konukoglu, 2018] and density-based models [Schlegl et al., 2017, Deecke et al., 2018].

All these approaches assume a training dataset of "normal" data. However, in many practical scenarios there will be unlabeled anomalies hidden in the training data. Wang et al. [2019], Huyan et al. [2021] have shown that anomaly detection accuracy deteriorates when the training set is contaminated. Our work provides a training strategy to deal with contamination.

Anomaly Detection on contaminated training data. 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* [Wang et al., 2019]. Throughout our paper, we refer to the strategy of blindly training an anomaly detector as if the training data was clean as "*Blind*" training. Yoon et al. [2021] 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 to continue training on the refined dataset. Similar data refinement strategy are also combined with latent SVDD [Görnitz et al., 2014] or autoencoders for anomaly detection [Xia et al., 2015, Beggel et al., 2019]. However, these methods fail to exploit the insight of outlier exposure [Hendrycks et al., 2018] that anomalies provide a valuable training signal. Zhou and Paffenroth [2017] used a robust autoencoder for identifying anomalous training data points, but their approach requires training a new model for identifying anomalies, which is

impractical in most setups. Hendrycks et al. [2018] propose to artificially contaminate the training data with samples from a related domain which can then be considered anomalies. While outlier exposure assumes labeled anomalies, our work aims at exploiting unlabeled anomalies in the training data. Notably, Pang et al. [2020] have used an iterative scheme to detect abnormal frames in video clips, and Feng et al. [2021a] extend it to supervised video anomaly detection. Our work is more general and provides a principled way to improve the training strategy of all approaches mentioned in the paragraph "deep anomaly detection" when the training data is likely contaminated.

3.3 Method

We will start by describing the mathematical foundations of our method. We will then describe our learning algorithm as a block coordinate descent algorithm, providing a theoretical convergence guarantee. Finally, we describe how our approach is applicable in the context of various state-of-the-art deep anomaly detection methods.

3.3.1 Problem Formulation

Setup. In this paper, we study the problem of unsupervised anomaly detection with contaminated training data. We consider a data set of samples $\mathbf{x}_i$; these could either come from a data distribution of "normal" samples, or could otherwise come from an unknown corruption process and thus be considered as "anomalies". For each datum $\mathbf{x}_i$, let $y_i = 0$ if the datum is normal, and $y_i = 1$ if it is anomalous. We assume that these binary labels are unobserved, both in our training and test sets, and have to be inferred from the data.

In contrast to most anomaly detection setups where the training dataset comprises only normal data, 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 contamination ratio α as a hyperparameter in our approach and denote α_0 as the ground truth contamination ratio where needed. Note that an assumed contamination ratio is a common hyperparameter in many robust algorithms [e.g., Huber, 1992, 2011], and we test the robustness of our approach w.r.t. this parameter in Section 3.4.

Our goal is to train a (deep) anomaly detection classifier on such corrupted data based on self-supervised or unsupervised training paradigms (see related work). The challenge thereby is to simultaneously infer the binary labels y_i during training while optimally exploiting this information for training an anomaly detection model.

Proposed Approach. We consider two losses. Similar to most work on deep anomaly detection, we consider a loss function $\mathcal{L}_n^\theta(\mathbf{x}) \equiv \mathcal{L}_n(f_\theta(\mathbf{x}))$ that we aim to minimize over "normal" data. The function $f_\theta(\mathbf{x})$ is used to extract features from $\mathbf{x}$, typically based on a self-supervised auxiliary task, see Section 3.3.4 for examples. 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.

In addition, we also consider a second loss for anomalies $\mathcal{L}_a^\theta(\mathbf{x}) \equiv \mathcal{L}_a(f_\theta(\mathbf{x}))$ (the feature extractor $f_\theta(\mathbf{x})$ is 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 $\mathcal{L}_n^\theta(\mathbf{x})$. For example, if $\mathcal{L}_n^\theta(\mathbf{x}) = ||f_\theta(\mathbf{x}) - \mathbf{c}||^2$ as in Deep SVDD [Ruff et al., 2018] (thus pulling normal data points towards their center), we define $\mathcal{L}_a^\theta(\mathbf{x}) = 1/||f_\theta(\mathbf{x}) - \mathbf{c}||^2$ (pushing abnormal data away from it) as in [Ruff et al., 2019].

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

$$\mathcal{L}(\theta, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^{N} (1 - y_i) \mathcal{L}_n^\theta(\mathbf{x}_i) + y_i \mathcal{L}_a^\theta(\mathbf{x}_i). \tag{3.1}$$

This equation resembles the log-likelihood of a probabilistic mixture model, but note that $\mathcal{L}_n^\theta(\mathbf{x}_i)$ and $\mathcal{L}_a^\theta(\mathbf{x}_i)$ are not necessarily data log-likelihoods; rather, self-supervised auxiliary losses can be used and often perform better in practice [Ruff et al., 2018, Qiu et al., 2021, Nalisnick et al., 2018].

Optimizing Eq. 3.1 over its parameters θ yields a better anomaly detector than $\mathcal{L}_n^\theta$ trained in isolation. By construction of the anomaly loss $\mathcal{L}_a^\theta$, the known anomalies provide an additional training signal to $\mathcal{L}_n^\theta$: due to parameter sharing, the labeled anomalies teach $\mathcal{L}_n^\theta$ where *not* to expect normal data in feature space. This is the basic idea of Outlier Exposure [Hendrycks et al., 2018], which constructs artificial *labeled* anomalies for enhanced detection performance.

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

3.3.2 Optimization problem

"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 y_i. Due to our assumption of having a fixed rate of anomalies α in the training data, we introduce a

constrained set:

$$\mathcal{Y} = \{\mathbf{y} \in \{0,1\}^N : \sum_{i=1}^{N} y_i = \alpha N\}. \tag{3.2}$$

The set describes a "hard" label assignment; hence the name "Hard LOE", which is the default version of our approach. Section 3.3.3 describes an extension with "soft" label assignments. Note that we require αN to be an integer.

Since our goal is to use the losses $\mathcal{L}_n^\theta$ and $\mathcal{L}_a^\theta$ to identify and score anomalies, we seek $\mathcal{L}_n^\theta(\mathbf{x}_i) - \mathcal{L}_a^\theta(\mathbf{x}_i)$ to be large for anomalies, and $\mathcal{L}_a^\theta(\mathbf{x}_i) - \mathcal{L}_n^\theta(\mathbf{x}_i)$ to be large for normal data. Assuming these losses to be optimized over θ, our best guess to identify anomalies is to minimize Equation (3.1) over the assignment variables $\mathbf{y}$. Combining this with the constraint (Equation (3.2)) yields the following minimization problem:

$$\min_\theta \min_{\mathbf{y}\in\mathcal{Y}} \mathcal{L}(\theta, \mathbf{y}). \tag{3.3}$$

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 $\mathbf{y}^t$ and proceed with alternating updates. To update θ, we simply fix $\mathbf{y}^t$ and minimize $\mathcal{L}(\theta, \mathbf{y}^t)$ over θ. In practice, we perform a single gradient step (or stochastic gradient step, see below), yielding a partial update.

To update $\mathbf{y}$ given θ^t, we minimize the same function subject to the constraint (Equa-

tion (3.2)). To this end, we define training anomaly scores,

$$S_i^{train} = \mathcal{L}_n^\theta(\mathbf{x}_i) - \mathcal{L}_a^\theta(\mathbf{x}_i). \tag{3.4}$$

These scores quantify the effect of y_i on minimizing Equation (3.1). We rank these scores and assign the $(1-\alpha)$-quantile of the associated labels 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 anomalies α in our experiments section. We stress that our testing anomaly scores will be different (see Section 3.3.3).

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 (3.1) based on mini-batches. For simplicity and memory efficiency, we impose the label constraint Equation (3.2) on each mini-batch and optimize θ and $\mathbf{y}$ in the same alternating fashion. The induced bias vanishes for large mini-batches. In practice, we found that this approach leads to satisfying results[2].

Algorithm 1 summarizes our approach.

3.3.3 Model extension and anomaly detection

We first discuss an important extension of our approach and then discuss its usage in anomaly detection.

[2]Note that an exact mini-batch version of the optimization problem in Equation (3.3) would also be possible, requiring memorization of $\mathbf{y}$ for the whole data set.

Algorithm 1: Training process of LOE

Input: Contaminated training set $\mathcal{D}$ (α_0 anomaly rate)
hyperparamter α
Model: Deep anomaly detector with parameters θ
foreach *Epoch* **do**
  foreach *Mini-batch* $\mathcal{M}$ **do**
    Calculate the anomaly score S_i^{train} for $\mathbf{x}_i \in \mathcal{M}$
    Estimate the label y_i given S_i^{train} and α
    Update the parameters θ by minimizing $\mathcal{L}(\theta, \mathbf{y})$
  end
end

"Soft" Latent Outlier Exposure (LOE$_S$). In practice, the block coordinate descent procedure can be overconfident in assigning $\mathbf{y}$, 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:

$$\mathcal{Y}' = \{\mathbf{y} \in \{0, 0.5\}^N : \sum_{i=1}^{N} y_i = 0.5\alpha N\}. \tag{3.5}$$

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

The consequence of an identified anomaly $y_i = 0.5$ is that we minimize an equal combination of both losses, $0.5(\mathcal{L}_n^\theta(\mathbf{x}_i) + \mathcal{L}_a^\theta(x_i))$. The interpretation is that the algorithm is uncertain about whether to treat $\mathbf{x}_i$ as a normal or anomalous data point and treats both cases as equally likely. A similar weighting scheme has been proposed for supervised learning in the presence of unlabeled examples [Lee and Liu, 2003]. In practice, we found the soft scheme to sometimes outperform the hard one (see Section 3.4).

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 3.3.2. 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

$\mathcal{L}_a^\theta$ during testing and score anomalies using only $\mathcal{L}_n^\theta$. Note that due to parameter sharing, training $\mathcal{L}_a^\theta$ jointly with $\mathcal{L}_n^\theta$ has already led to the desired information transfer between both losses.

Testing is the same for both "soft" LOE (Section 3.3.2) and "hard" LOE (Section 3.3.3). We define our testing anomaly score in terms of the "normal" loss function,

$$S_i^{test} = \mathcal{L}_n^\theta(\mathbf{x}_i). \tag{3.6}$$

3.3.4 Example loss functions

As follows, we review several loss functions that are compatible with our approach. We consider three advanced classes of self-supervised anomaly detection methods. These methods are i) MHRot [Hendrycks et al., 2019], ii) NTL [Qiu et al., 2021], and iii) ICL [Shenkar and Wolf, 2022]. While no longer being considered as a competitive baseline, we also consider deep SVDD for visualization due to its simplicity.

Multi-Head Rotation Net (MHRot) [Hendrycks et al., 2019] amounts to learning a multi-head classifier to predict whether an image is augmented by handcrafted transformations such as rotations, horizontal shifts, and vertical shifts. Figure 2.2 provides some common examples of image augmentations used in self-supervised learning. Suppose M different transformations are available. Each image can be applied with multiple transformations, giving various augmentations as a result of compositions of transformations. Since each transformation can be added or not independently, the number of transformation compositions is $K = 2^M$. Denote these compositions by $\{T_k\}_{k=1}^K$. The multi-head classifier has M prediction heads with parameters θ_m, each representing $p(t_{k,m}|T_k(\mathbf{x}); \theta_m)$, where $t_{k,m} \in \{0, 1\}$ indicates whether or not a transformation m is used[3] in T_k. Aiming to predict the correct

[3]In practice, multi-class classification is also possible for each head. For example, various rotation angles

transformations for normal samples, we minimize the negative log-likelihoods of the ground truth label $t_{k,m}$ for each transformation m and for each transformation composition T_k, resulting in the loss function for each data point $\mathbf{x}$

$$\mathcal{L}^{\theta}_{\text{MHRot}}(\mathbf{x}) := -\sum_{k=1}^{K}\sum_{m=1}^{M}\log p(t_{k,m}|T_k(\mathbf{x});\theta_m). \tag{3.7}$$

The loss function on the training dataset is $\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}^{\theta}_{\text{MHRot}}(\mathbf{x}_i)$. At test time, the anomaly score is the same as the loss function $S(\mathbf{x};\theta) = \mathcal{L}^{\theta}_{\text{MHRot}}(\mathbf{x})$, or the entropy[4] of the model predictions $S(\mathbf{x};\theta) = \sum_{k=1}^{K}\sum_{m=1}^{M} H(p(\cdot|T_k(\mathbf{x});\theta_m))$. Since abnormal data is not observed during training, the anomaly score is expected to be higher for anomalous data than normal data at test time.

Neural Transformation Learning (NTL). Rather than using hand-crafted transformations, NTL learns K neural transformations $\{T_{\theta,1}, ..., T_{\theta,K}\}$ and an encoder f_θ parameterized by θ from data and uses the learned transformations to detect anomalies. Each neural transformation generates a view $\mathbf{x}_k = T_{\theta,k}(\mathbf{x})$ of sample $\mathbf{x}$. For normal samples, NTL encourages each transformation to be similar to the original sample and to be dissimilar from other transformations. To achieve this objective, NTL maximizes the normalized probability $p_k = h(\mathbf{x}_k, \mathbf{x})\big/\big(h(\mathbf{x}_k, \mathbf{x}) + \sum_{l\neq k} h(\mathbf{x}_k, \mathbf{x}_l)\big)$ for each view where $h(\mathbf{a}, \mathbf{b}) = \exp(\cos(f_\theta(\mathbf{a}), f_\theta(\mathbf{b}))/\tau)$ measures the similarity of two views [5]. For anomalies, we "flip" the objective for normal samples: the model instead pulls the transformations close to each other and pushes them away from the original view, resulting in

$$\mathcal{L}^{\theta}_{n}(\mathbf{x}) := -\sum_{k=1}^{K}\log p_k, \quad \mathcal{L}^{\theta}_{a}(\mathbf{x}) := -\sum_{k=1}^{K}\log(1-p_k).$$

can be applied to augment an image, and then the head predicts which angle is applied.

[4]The entropy of a discrete probability distribution is $H(p(x)) = -\sum_x p(x)\log p(x)$.

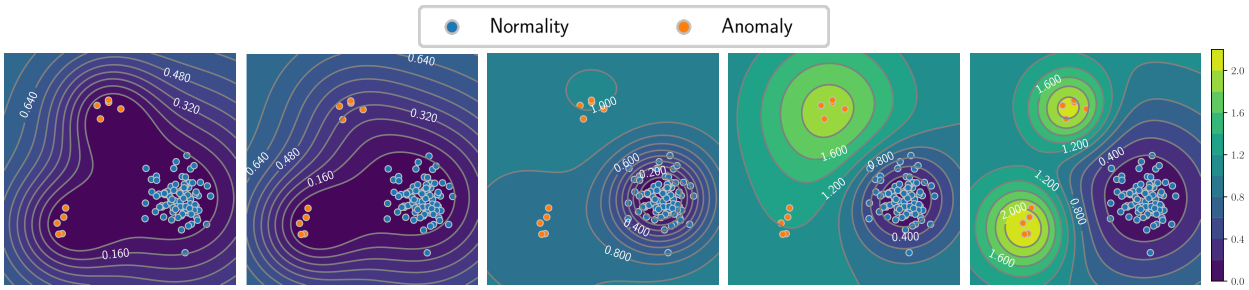
[5]where τ is the temperature and $\cos(\mathbf{a}, \mathbf{b}) := \mathbf{a}^T\mathbf{b}/\|\mathbf{a}\|\|\mathbf{b}\|$

Internal Contrastive Learning (ICL). ICL is a state-of-the-art *tabular* anomaly detection method [Shenkar and Wolf, 2022]. 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, $a(\mathbf{x})$ and $b(\mathbf{x})$, in the embedding space. The maximization of the mutual information is equivalent to minimizing a contrastive loss $\mathcal{L}_n^\theta(\mathbf{x}) := -\sum_{k=1}^K \log p_k$ on normal samples with $p_k = h(a_k(\mathbf{x}), b_k(\mathbf{x})) \big/ \sum_{l=1}^K h(a_l(\mathbf{x}), b_k(\mathbf{x}))$ where $h(a, b) = \exp(\cos(f_\theta(a), g_\theta(b))/\tau)$ measures the similarity of two feature subsets in the embedding space of two encoders f_θ and g_θ. For anomalies, we flip the objective as $\mathcal{L}_a^\theta(\mathbf{x}) := -\sum_{k=1}^K \log(1 - p_k)$.

3.4 Experiments

We evaluate our proposed methods and baselines for unsupervised anomaly detection tasks on different data types: synthetic data, tabular data, images, and videos. The data are contaminated with different anomaly ratios. Depending on the data, we study our method in combination with specific backbone models. MHRot applies only to images and ICL to tabular data. NTL can be applied to all data types.

We have conducted extensive experiments on image, tabular, and video data. For instance, we evaluate our methods on all 30 tabular datasets of Shenkar and Wolf [2022]. Our proposed method sets a new state-of-the-art on most datasets. In particular, we show that our method gives robust results even when the contamination ratio is unknown.



(a) Blind (b) Refine (c) LOE_S (d) LOE_H (e) G-truth

Figure 3.1: Deep SVDD trained on 2D synthetic contaminated data (see main text) trained with different methods: **(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.

3.4.1 Toy Example

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 3.1). For simplicity, the backbone anomaly detector is the deep one-class classifier [Ruff et al., 2018] with radial basis functions. Setting the contamination ratio to $\alpha_0 = \alpha = 0.1$, we compare the baselines "Blind" and "Refine" (described in Section 3.2, detailed in Appendix B.2) with the proposed LOE_H and LOE_S (described in Section 3.3) and the theoretically optimal *G-truth* method (which uses the ground truth labels). We defer all further training details to Appendix B.1.

Figure 3.1 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 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.

3.4.2 Experiments on Image Data

Anomaly detection on images is especially far developed. We demonstrate LOE's benefits when applied to two leading image anomaly detectors as backbone models: MHRot and 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: CIFAR-10, Fashion-MNIST, and MVTEC [Bergmann et al., 2019]. These have been used in virtually all deep anomaly detection papers published at top-tier venues [Ruff et al., 2018, Golan and El-Yaniv, 2018, Hendrycks et al., 2019, Bergman and Hoshen, 2020, Li et al., 2021b], and we adopt these papers' experimental protocol here, as detailed below.

Backbone models and baselines. We experiment with MHRot and NTL. In consistency with previous work [Hendrycks et al., 2019], we train MHRot on raw images and NTL on features outputted by an encoder pre-trained on ImageNet. We use the official code by the respective authors[6][7]. NTL is built upon the final pooling layer of a pre-trained ResNet152 for CIFAR-10 and F-MNIST (as suggested in Defard et al. [2021]), and upon the third residual block of a pre-trained WideResNet50 for MVTEC (as suggested in Reiss et al. [2021]). Further implementation details of NTL are in the Appendix B.3.

Many existing baselines apply either blind updates or a refinement strategy to specific backbone models (see Section 3.2). However, a recent study showed that many of the classical anomaly detection methods such as autoencoders are no longer on par with modern self-

[6] `https://github.com/hendrycks/ss-ood.git`
[7] `https://github.com/boschresearch/NeuTraL-AD.git`

Table 3.1: AUC (%) with standard deviation for anomaly detection on CIFAR-10 and F-MNIST. For all experiments, we set the contamination ratio as 10%. LOE mitigates the performance drop when NTL and MHRot trained on the contaminated datasets.

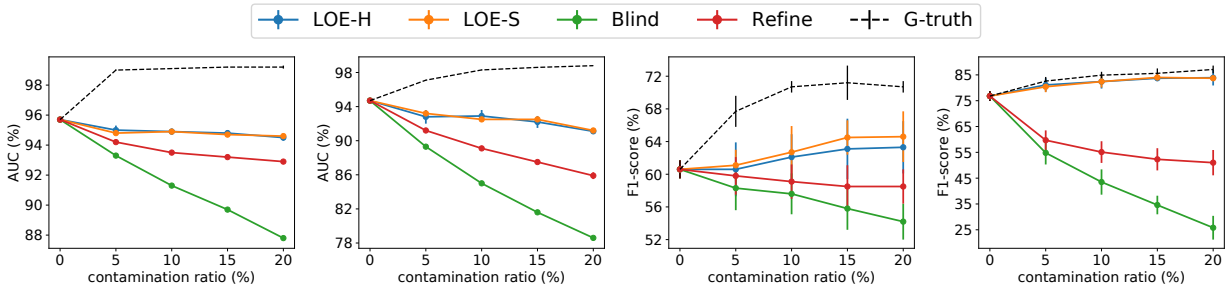
		CIFAR-10	F-MNIST
NTL	Blind	91.3±0.1 (-4.4)	85.0±0.2 (-9.7)
	Refine	93.5±0.1 (-2.2)	89.1±0.2 (-5.6)
	LOE_H (ours)	**94.9±0.2 (-0.8)**	**92.9±0.7 (-1.8)**
	LOE_S (ours)	**94.9±0.1 (-0.8)**	92.5±0.1 (-2.2)
MHRot	Blind	84.0±0.5 (-4.2)	88.8±0.1 (-4.9)
	Refine	84.4±0.1 (-3.8)	89.6±0.2 (-4.1)
	LOE_H (ours)	**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)

supervised approaches [Alvarez et al., 2022, Hendrycks et al., 2019] and in particular found NTL to perform best among 13 considered models. 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 3.3) and the two baseline methods "Blind" and "Refine" (Section 3.2) to two backbone models.

Image datasets. On CIFAR-10 and F-MNIST, we follow the standard "one-vs.-rest" protocol of converting these data into anomaly detection datasets [Ruff et al., 2018, Golan and El-Yaniv, 2018, Hendrycks et al., 2019, Bergman and Hoshen, 2020]. We create C anomaly detection tasks (where C 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 image 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. [2021]. 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.

Table 3.2: AUC (%) with standard deviation of NTL for anomaly detection/segmentation on MVTEC. We set the contamination ratio of the training set as 10% and 20%.

	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 (ours)	**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 (ours)	95.4±0.5 (-2.0)	**93.6±0.3** (-3.8)	**96.56±0.04** (-0.39)	**96.11±0.05** (-0.84)



(a) CIFAR-10 (b) F-MNIST (c) Arrhythmia (d) Thyroid

Figure 3.2: Anomaly detection performance of NTL on CIFAR-10, F-MNIST, and two tabular datasets (Arrhythmia and Thyroid) with $\alpha_0 \in \{5\%, 10\%, 15\%, 20\%\}$. LOE (ours) consistently outperforms the "Blind" and "Refine" on various contamination ratios.

Results. We present the experimental results of CIFAR-10 and F-MNIST in Table 3.1, 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. Our proposed methods consistently outperform the baselines and mitigate the performance drop between the model trained on clean data vs. the same model trained on contaminated data. Specifically, with NTL, LOE significantly improves over the best-performing baseline, "Refine", by 1.4% and 3.8% AUC on CIFAR-10 and F-MNIST, respectively. On CIFAR-10, our methods have only 0.8% AUC lower than when training on the normal dataset. When we use another state-of-the-art method MHRot on raw images, our LOE methods outperform the baselines by about 2% AUC on both

Table 3.3: F1-score (%) for anomaly detection on 30 tabular datasets studied in [Shenkar and Wolf, 2022]. We set $\alpha_0 = \alpha = 10\%$ in all experiments. LOE (proposed) outperforms the "Blind" and "Refine" consistently. (See Tables B.1 and B.2 for more details, including AUCs.)

	NTL				ICL			
	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)
abalone	37.9±13.4	55.2±15.9	42.8±26.9	**59.3±12.0**	50.9±1.5	**54.3±2.9**	53.4±5.2	51.7±2.4
annthyroid	29.7±3.5	42.7±7.1	47.7±11.4	**50.3±4.5**	29.1±2.2	38.5±2.1	**48.7±7.6**	43.0±8.8
arrhythmia	57.6±2.5	59.1±2.1	62.1±2.8	**62.7±3.3**	53.9±0.7	60.9±2.2	62.4±1.8	**63.6±2.1**
breastw	84.0±1.8	93.1±0.9	**95.6±0.4**	95.3±0.4	92.6±1.1	93.4±1.0	**96.0±0.6**	95.7±0.6
cardio	21.8±4.9	45.2±7.9	**73.0±7.9**	57.8±5.5	50.2±4.5	56.2±3.4	**71.1±3.2**	62.2±2.7
ecoli	0.0±0.0	88.9±14.1	**100±0.0**	**100±0.0**	17.8±15.1	46.7±25.7	**75.6±4.4**	**75.6±4.4**
forest cover	20.4±4.0	56.2±4.9	61.1±34.9	**67.6±30.6**	9.2±4.5	8.0±3.6	6.8±3.6	**11.1±2.1**
glass	11.1±7.0	15.6±5.4	17.8±5.4	**20.0±8.3**	8.9±4.4	**11.1±0.0**	**11.1±7.0**	8.9±8.3
ionosphere	89.0±1.5	91.0±2.0	91.0±1.7	**91.3±2.2**	86.5±1.1	85.9±2.3	85.7±2.8	**88.6±0.6**
kdd	95.9±0.0	96.0±1.1	98.1±0.4	**98.4±0.1**	99.3±0.1	99.4±0.1	**99.5±0.0**	99.4±0.0
kddrev	98.4±0.1	98.4±0.2	89.1±1.7	**98.6±0.0**	97.9±0.5	98.4±0.4	**98.8±0.1**	98.2±0.4
letter	36.4±3.6	44.4±3.1	25.4±10.0	**45.6±10.6**	43.0±2.5	51.2±3.7	**54.4±5.6**	47.2±4.9
lympho	53.3±12.5	60.0±8.2	60.0±13.3	**73.3±22.6**	43.3±8.2	60.0±8.2	80.0±12.5	**83.3±10.5**
mammogra.	5.5±2.8	2.6±1.7	3.3±1.6	**13.5±3.8**	8.8±1.9	11.4±1.9	34.0±20.2	**42.8±17.6**
mnist tabular	78.6±0.5	**80.3±1.1**	71.8±1.8	76.3±2.1	72.1±1.0	80.7±0.7	**86.0±0.4**	79.2±0.9
mulcross	45.5±9.6	**58.2±3.5**	**58.2±6.2**	50.1±8.9	70.4±13.4	94.4±6.3	**100±0.0**	99.9±0.1
musk	21.0±3.3	98.8±0.4	**100±0.0**	**100±0.0**	6.2±3.0	**100±0.0**	**100±0.0**	**100±0.0**
optdigits	0.2±0.3	1.5±0.3	41.7±45.9	**59.1±48.2**	0.8±0.5	**1.3±1.1**	1.2±1.0	0.9±0.5
pendigits	5.0±2.5	32.6±10.0	79.4±4.7	**81.9±4.3**	10.3±4.6	30.1±8.5	80.3±6.1	**88.6±2.2**
pima	60.3±2.6	61.0±1.9	**61.3±2.4**	61.0±0.9	58.1±2.9	59.3±1.4	**63.0±1.0**	60.1±1.4
satellite	73.6±0.4	74.1±0.3	**74.8±0.4**	74.7±0.1	72.7±1.3	72.7±0.6	**73.6±0.2**	73.2±0.6
satimage	26.8±1.5	86.8±4.0	90.7±1.1	**91.0±0.7**	7.3±0.6	85.1±1.4	91.3±1.1	**91.5±0.9**
seismic	11.9±1.8	11.5±1.0	**18.1±0.7**	17.1±0.6	14.9±1.4	17.3±2.1	23.6±2.8	**24.2±1.4**
shuttle	97.0±0.3	97.0±0.2	**97.1±0.2**	97.0±0.2	96.6±0.2	96.7±0.1	96.9±0.1	**97.0±0.2**
speech	6.9±1.2	8.2±2.1	43.3±5.6	**50.8±2.5**	0.3±0.7	1.6±1.0	**2.0±0.7**	0.7±0.8
thyroid	43.4±5.5	55.1±4.2	**82.4±2.7**	**82.4±2.3**	45.8±7.3	71.6±2.4	**83.2±2.9**	80.9±2.5
vertebral	22.0±4.5	21.3±4.5	22.7±11.0	**25.3±4.0**	8.9±3.1	8.9±4.2	7.8±4.2	**10.0±2.7**
vowels	36.0±1.8	50.4±8.8	**62.8±9.5**	48.4±6.6	42.1±9.0	60.4±7.9	**81.6±2.9**	74.4±8.0
wbc	25.7±12.3	45.7±15.5	**76.2±6.0**	69.5±3.8	50.5±5.7	50.5±2.3	**61.0±4.7**	**61.0±1.9**
wine	24.0±18.5	66.0±12.0	90.0±0.0	**92.0±4.0**	4.0±4.9	10.0±8.9	98.0±4.0	**100±0.0**

datasets.

We also evaluate our methods with NTL at various contamination ratios (from 5% to 20%) in Figure 3.2 (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 experimented on anomaly detection and segmentation on the MVTEC dataset. Results are shown in Table 3.2, 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.

3.4.3 Experiments on Tabular Data

Tabular data is another important application area of anomaly detection. Many data sets in the healthcare and cybersecurity domains are tabular. Our empirical 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 [Shenkar and Wolf, 2022]. 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[8]. We follow the pre-processing and train-test split of the datasets in Shenkar and Wolf [2022]. 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.

Backbone models and baselines. We consider two advanced deep anomaly detection methods for tabular data described in Section 3.3.4: NTL and ICL. For NTL, we use nine transformations and multi-layer perceptrons for neural transformations and the encoder on all datasets. Further details are provided in Appendix B.3. For ICL, we use the code provided by the authors. We implement the proposed LOE methods (Section 3.3) and the "Blind" and "Refine" baselines (Section 3.2) with both backbone models.

Results. We report F1-scores for 30 tabular datasets in Table 3.3. The results are reported as the mean and standard derivation of five runs with different model initializations and

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

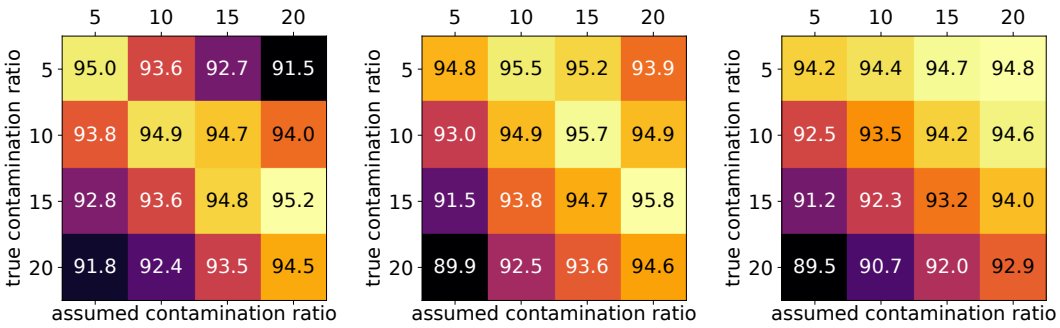
Table 3.4: AUC (%) for different contamination ratios for a video frame anomaly detection benchmark proposed in [Pang et al., 2020]. LOE_S (proposed) achieves state-of-the-art performance.

Method	Contamination Ratio		
	10%	20%	30%*
[Tudor Ionescu et al., 2017]	-	-	68.4
[Liu et al., 2018]	-	-	69.0
[Del Giorno et al., 2016]	-	-	59.6
[Sugiyama and Borgwardt, 2013]	55.0	56.0	56.3
[Pang et al., 2020]	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., 2020], corresponding to $\alpha_0 \approx 30\%$.

random training set split. We set the contamination ratio $\alpha_0 = \alpha = 0.1$ for all datasets. More detailed results, including AUCs and the performance degradation over clean data, are provided in Appendix B.4 (Tables B.1 and B.2).

LOE outperforms the "Blind" and "Refine" baselines consistently. Remarkably, on some datasets, LOE trained on contaminated data can achieve better results than on clean data (as shown in Table B.1), 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 3.2 (c) and (d)). Hendrycks et al. [2018] noticed a similar phenomenon when adding *labeled* auxiliary 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.



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

Figure 3.3: A sensitivity study of the robustness of LOE_H, LOE_S, and "Refine" to the misspecified contamination ratio. We evaluate them with NTL on CIFAR-10 in terms of AUC. LOE_H and LOE_S yield robust results and outperform "Refine" in the most cases.

3.4.4 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., 2020]. 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[9], 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. [2020] 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.

Backbone models and baselines. In addition to the "Blind" and "Refine" baselines, we compare to [Pang et al., 2020] (a ranking-based state-of-the-art method for video frame

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

anomaly detection already described in Section 3.2) and all baselines reported in that paper [Sugiyama and Borgwardt, 2013, Liu et al., 2012, Del Giorno et al., 2016, Tudor Ionescu et al., 2017, Liu et al., 2018].

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.

Results. We report the results in Table 3.4. Our soft LOE method achieves the best performance across different contamination ratios. Our method outperforms Deep Ordinal Regression [Pang et al., 2020] by 18.8% and 9.2% AUC on the contamination ratios of 10% and 20%, respectively. LOE$_S$ outperforms the "Blind" and "Refine" baselines significantly on various contamination ratios.

3.4.5 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$_H$ and LOE$_S$ 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 3.3 (a)) shows considerable robustness: the method suffers at most 1.4% performance degradation when the hyperparameter α is off by 5%, and is always better than "Blind". It always outperforms "Refine" (Figure 3.3 (c)) when erroneously setting a smaller α than the true ratio α_0. LOE$_S$ (Figure 3.3 (b)) also shows robustness, especially when erroneously setting a larger α than α_0. The method is always better than "Refine"

(Figure 3.3 (c)) when the hyperparameter α is off by up to 15%, and always outperforms "Blind".

3.5 Conclusion

We propose Latent Outlier Exposure (LOE): a domain-independent approach for training anomaly detectors on a dataset contaminated by unidentified anomalies. During training, LOE jointly infers anomalous data in the training set while updating its parameters by solving a mixed continuous-discrete optimization problem; iteratively updating the model and its predicted anomalies. Similar to outlier exposure [Hendrycks et al., 2018], LOE extracts a learning signal 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 three of image, tabular, and video data.

Chapter 4

Deep Anomaly Detection under Labeling Budget Constraints

This chapter is based on a published paper at ICML 2023: *Deep Anomaly Detection under Labeling Budget Constraints* by Aodong Li*, Chen Qiu*, Marius Kloft, Padhraic Smyth, Stephan Mandt, Maja Rudolph [Li et al., 2023]

4.1 Introduction

Detecting anomalies in data is a fundamental task in machine learning with applications across multiple domains, from industrial fault detection to medical diagnosis. The main idea is to train a model (such as a neural network) on a data set of "normal" samples to minimize the loss of an auxiliary (e.g., self-supervised) task. Using the loss function to score test data, one hopes to obtain low scores for normal data and high scores for anomalies [Ruff et al., 2021].

In practice, the training data is often contaminated with unlabeled anomalies that differ

in unknown ways from the i.i.d. samples of normal data. No access to a binary anomaly label (indicating whether a sample is normal or not) makes learning the anomaly scoring function from contaminated data challenging; the training signal has to come exclusively from the input features (typically real-valued vectors). Many approaches either assume that the unlabeled anomalies are too rarely encountered during training to affect learning [Wang et al., 2019] or try to detect and exploit the anomalies in the training data (e.g., Qiu et al. [2022b]).

While anomaly detection is typically an unsupervised training task, sometimes expert feedback is available to check if individual samples are normal or not. For example, in a medical setting, one may ask a medical doctor to confirm whether a given image reflects normal or abnormal cellular tissue. Other application areas include detecting network intrusions or machine failures. Anomaly labels are usually expensive to obtain but are very valuable to guide an anomaly detector during training. For example, in Figure 4.1, we can see that our method, with only one labeled query (Figure 4.1 d) is almost on par with supervised anomaly detection (Figure 4.1 a). However, the supervised setting is unrealistic, since expert feedback is typically expensive. Instead, it is essential to develop effective strategies for querying informative data points.

Previous work on anomaly detection under a labeling budget primarily involves domain-specific applications and/or ad hoc architectures, making it hard to disentangle modeling choices from querying strategies [Trittenbach et al., 2021]. In contrast, this chapter theoretically and empirically studies generalization performance using various labeling budgets, querying strategies, and losses.

In summary, our main contributions are as follows:

1. We prove that the ranking of anomaly scores generalizes from labeled queries to unlabeled data under certain conditions that characterize how well the queries cover the data. Based

on this theory, we propose a diverse querying strategy for deep anomaly detection under labeling budget constraints.

2. We propose SOEL, a semi-supervised learning framework compatible with a large number of deep anomaly detection losses. We show how all major hyperparameters can be eliminated, making SOEL easy to use. To this end, we provide an estimate for the anomaly ratio in the data.
3. We provide an extensive benchmark for deep anomaly detection with a limited labeling budget. Our experiments on image, tabular, and video data provide evidence that SOEL outperforms existing methods significantly. Comprehensive ablations disentangle the benefits of each component.

This chapter is structured as follows. Section 4.3 introduces the problem setting we address and our main algorithm. Section 4.2 discusses related work in deep anomaly detection. Section 4.4 discusses experimental results on each of image, video, and tabular data. Finally, we conclude this work in Section 4.5.

4.2 Related Work

Deep Anomaly Detection. Many recent advances in anomaly detection are in the area of deep learning [Ruff et al., 2021]. One early strategy was to use autoencoder- [Principi et al., 2017, Zhou and Paffenroth, 2017] or density-based models [Schlegl et al., 2017, Deecke et al., 2018]. Another pioneering stream of research combines one-class classification [Schölkopf et al., 2001] with deep learning for unsupervised [Ruff et al., 2018, Qiu et al., 2022a] and semi-supervised [Ruff et al., 2019] anomaly detection. Many other approaches to deep anomaly detection are self-supervised. They employ a self-supervised loss function to train the detector and score anomalies [Golan and El-Yaniv, 2018, Hendrycks et al., 2019, Bergman and Hoshen, 2020, Qiu et al., 2021, Shenkar and Wolf, 2022, Schneider et al., 2022].

Our work resides in the self-supervised anomaly detection category and can be extended to other data modalities if an appropriate loss is provided.

While all these methods assume that the training data consists of only normal samples, in many practical applications, the training pool may be contaminated with unidentified anomalies [Vilhjálmsson and Nordborg, 2013, Steinhardt et al., 2017]. This can be problematic because the detection accuracy typically deteriorates when the contamination ratio increases [Wang et al., 2019]. Addressing this, refinement [Zhou and Paffenroth, 2017, Yoon et al., 2021] attempts to cleanse the training pool by removing anomalies therein, although they may provide valuable training signals. As a remedy, Qiu et al. [2022b] propose to jointly infer binary labels to each datum (normal vs. anomalous) while updating the model parameters based on outlier exposure. Our work also makes the contaminated data assumption and employs the training signal of abnormal data.

Querying Strategies for Anomaly Detection. Querying strategies play an important role in batch active learning [Sener and Savarese, 2018, Ash et al., 2020, Citovsky et al., 2021, Pinsler et al., 2019, Hoi et al., 2006] but are less studied for anomaly detection. The human-in-the-loop setup for anomaly detection has been pioneered by Pelleg and Moore [2004]. Query samples are typically chosen locally, e.g., close to the decision boundary of a one-class SVM [Görnitz et al., 2013, Yin et al., 2018] or sampled according to a density model [Ghasemi et al., 2011]. Siddiqui et al. [2018], Das et al. [2016] propose to query the most anomalous instance, while Das et al. [2019] employ a tree-based ensemble to query both anomalous and diverse samples. A recent survey compares various aforementioned query strategies with one-class classifiers [Trittenbach et al., 2021].

Pimentel et al. [2020] query samples with the top anomaly scores for autoencoder-based methods, while Ning et al. [2022] improve the querying by considering the diversity. Tang et al. [2020] use an ensemble of deep anomaly detectors and query the most likely anomalies for each detector separately. Russo et al. [2020] query samples where the model is uncertain

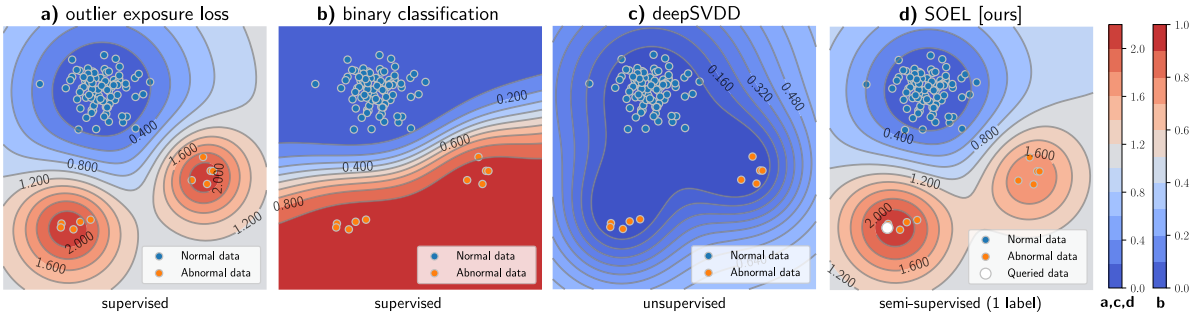


Figure 4.1: Anomaly score contour plots on 2D toy data demonstrate that SOEL [ours, (**d**)] with only one labeled sample can achieve detection accuracy that is competitive with a fully supervised approach (**a**). Binary classification (**b**) is problematic for anomaly detection since it cannot detect new anomalies, e.g. in the upper right corner of the plot. Subplot (**c**) demonstrates that unsupervised anomaly detection is challenging with contaminated data. Even a single labeled query, in combination with our approach, can significantly improve anomaly detection.

about the predictions. Pang et al. [2021b] and Zha et al. [2020] propose querying strategies based on reinforcement learning, which requires labeled datasets.

All these querying strategies do not optimize coverage as defined in Theorem 4.1, and as a result, their generalization guarantees are less favorable than our method. Most querying strategies from the papers discussed above are fairly general and can be applied in combination with various backbone models. Since more powerful backbone models have been released since these earlier publications, we ensure a fair comparison by studying all querying strategies in combination with the same backbone models as SOEL.

4.3 Methods

4.3.1 Notation and Problem Statement

Consider a dataset $\{\mathbf{x}_i\}_{i=1}^N$ where the datapoints $\mathbf{x}_i$ are i.i.d. samples from a mixture distribution $p(\mathbf{x}) = (1-\alpha)p_n(\mathbf{x}) + \alpha p_a(\mathbf{x})$. The distribution $p_n(\mathbf{x})$ corresponds to the normal data, while $p_a(\mathbf{x})$ corresponds to anomalous data. We assume that $0 \leq \alpha < 0.5$, i.e., that

the anomalous data is non-dominant in the mixture; in practice, $\alpha \ll 0.5$.

In the anomaly detection problem, we wish to use the data to train an anomaly detector in the form of a parametric anomaly score function $S(\mathbf{x}; \theta)$. Once trained this score function is thresholded to determine whether a datapoint $\mathbf{x}_i$ is anomalous, as indicated by the binary anomaly label $y_i := y(\mathbf{x}_i) \in \{0 := \text{“normal”}, 1 := \text{“abnormal”}\}$.

We focus on the situation where the training data is unlabeled (only $\mathbf{x}_i$ is known, not y_i), but where we have access to an oracle (e.g., a human expert) that is able to provide labels y_i for a budgeted number K of the N training points.

4.3.2 Outline of the Technical Approach

Our work addresses the following questions: How to best select informative data points for labeling – this is called the *querying strategy*, how to best learn an anomaly detector from both the labeled and unlabeled data in a semi-supervised fashion, and how to make the approach easy to use by eliminating a crucial hyper-parameter.

Querying Strategy. A successful approach for deep anomaly detection under labeling budget constraints will require a strategy for selecting the most beneficial set of queries. We choose a theoretically-grounded approach based on generalization performance. For this, we exploit that at test-time an anomaly detection method will threshold the anomaly scores to distinguish between normal samples and anomalies. This means that the quality of a scoring function is not determined by the absolute anomaly scores but only by their relative ranking. In Section 4.3.4, we characterize a favorable property of the query set which can guarantee that the ranking of anomaly scores generalizes from the labeled data to unlabeled samples. Since this is desirable, we derive a querying strategy that under a limited labeling budget best fulfills the favorable properties put forth by our analysis.

Semi-supervised Outlier Exposure. As a second contribution, we propose a semi-supervised learning framework that best exploits both the labeled query set and the unlabeled data. It builds on supervised anomaly detection and LOE which we review in Section 4.3.3. We present SOEL in Section 4.3.5. The SOEL training objective is designed to receive opposing training signals from the normal samples and the anomalies. An EM-style algorithm alternates between estimating the anomaly labels of the unlabaled data and improving the anomaly scoring function using the data samples and their given or estimated labels.

Hyperparameter Elimination. Like related methods discussed in Section 3.2, SOEL has an important hyperparameter α which corresponds to the expected fraction of anomalies in the data. While previous work has to assume that α is known [Qiu et al., 2022b], our proposed method presents an opportunity to estimate it. The estimate has to account for the fact that the optimal querying strategy derived from our theory in Section 4.3.4 is not i.i.d.. In Section 4.3.6, we provide an estimate of α for any stochastic querying strategy.

4.3.3 Background: Deep Anomaly Detection

In deep anomaly detection, auxiliary losses help learn the anomaly scoring function $S(\mathbf{x}; \theta)$ (we also write $S(\mathbf{x})$ for simplicity when model parameters θ are not of interest). Popular losses include autoencoder-based losses [Zhou and Paffenroth, 2017], the deep SVDD loss [Ruff et al., 2018], or the neural transformation learning loss [Qiu et al., 2021]. It is assumed that minimizing such a loss $\mathcal{L}_n^\theta(\mathbf{x}) \equiv \mathcal{L}_n(S(\mathbf{x}; \theta))$ over "normal" data leads to a desirable scoring function that assigns low scores to normal samples and high scores to anomalies.

Most deep anomaly detection methods optimize such an objective over an entire unlabeled data set, even if it contains unknown anomalies. It is assumed that the anomalies are rare enough that they will not dilute the training signal provided by the normal samples (*inlier priority*, [Wang et al., 2019]). Building on the ideas of Ruff et al. [2019] that synthetic

anomalies can provide valuable training signal, Qiu et al. [2022b] show how to discover and exploit anomalies by treating the anomaly labels as latent variables in training.

The key idea of Ruff et al. [2019] is to construct a complementary loss $\mathcal{L}_a^\theta(\mathbf{x}) \equiv \mathcal{L}_a(S(\mathbf{x};\theta))$ for anomalies that has an opposing effect to the normal loss $\mathcal{L}_n^\theta(\mathbf{x})$. For example, the deep SVDD loss $\mathcal{L}_n^\theta(\mathbf{x}) = ||f_\theta(\mathbf{x}) - \mathbf{c}||^2$, with feature extractor f_θ, pulls normal data points towards a fixed center $\mathbf{c}$ [Ruff et al., 2018]. The opposing loss for anomalies, defined in Ruff et al. [2019] as $\mathcal{L}_a^\theta(\mathbf{x}) = 1/\mathcal{L}_n^\theta(\mathbf{x})$, pushes abnormal data away from the center.

Supervised anomaly detection. Using only the labeled data indexed by $\mathcal{Q}$ one could train $S(\mathbf{x};\theta)$ using a supervised loss [Hendrycks et al., 2018, Görnitz et al., 2013]

$$\mathcal{L}_\mathcal{Q}(\theta) = \frac{1}{|\mathcal{Q}|} \sum_{j \in \mathcal{Q}} \big(y_j \mathcal{L}_a^\theta(\mathbf{x}_j) + (1 - y_j) \mathcal{L}_n^\theta(\mathbf{x}_j) \big). \tag{4.1}$$

Latent Outlier Exposure. Latent outlier exposure (LOE, [Qiu et al., 2022b]) is an unsupervised anomaly detection framework that uses the same loss as Equation (4.1) but treats the labels y as latent variables. An EM-style algorithm alternates between optimizing the model w.r.t. θ and inferring the labels y.

In this work, we propose semi-supervised outlier exposure with a limited labeling budget (SOEL) which builds on these ideas. We next present the querying strategy and when the querying strategy leads to correct ranking of anomaly scores (Section 4.3.4), the SOEL loss (Section 4.3.5), and how the hyperparameter α can be eliminated (Section 4.3.6)

4.3.4 Querying Strategies for Anomaly Detection

The first ingredient of SOEL is a querying strategy for selecting informative data points to be labeled, which we derive from theoretical considerations. An important property of the querying strategy is how well it covers unlabeled data. The quality of a querying strategy is

determined by the smallest radius δ such that all unlabeled points are within distance δ of one queried sample of the same type. In this paper, we prove that if the queries cover both the normal data and the anomalies well (i.e., if δ is small), a learned anomaly detector that satisfies certain conditions is guaranteed to generalize correctly to the unlabeled data (The exact statement and its conditions will be provided in Theorem 4.1). Based on this insight, we propose to use a querying strategy that is better suited for deep anomaly detection than previous work.

Theorem 4.1. *Let $\mathcal{Q}_0$ be the index set of datapoints labeled normal and $\mathcal{Q}_1$ the index set of datapoints labeled abnormal. Let $\delta \in \mathbb{R}^+$ be the smallest radius, such that for each unlabeled anomaly $\mathbf{u}_a$ and each unlabeled normal datum $\mathbf{u}_n$ there exist labeled data points $\mathbf{x}_a, a \in \mathcal{Q}_1$ and $\mathbf{x}_n, n \in \mathcal{Q}_0$, such that $\mathbf{u}_a$ is within the δ-ball of $\mathbf{x}_a$ and $\mathbf{u}_n$ is within the δ-ball around $\mathbf{x}_n$. If a λ_s-Lipschitz continuous function S ranks the labeled data correctly, with a large enough margin, i.e. $S(\mathbf{x}_a) - S(\mathbf{x}_n) \geq 2\delta\lambda_s$, then S ranks the unlabeled points correctly, too, and $S(\mathbf{u}_a) \geq S(\mathbf{u}_n)$.*

In Appendix C.1, we prove Theorem 4.1 and discuss the assumptions. An implication of this theorem is that a smaller δ corresponding to a tighter cover of the data leads to better-generalized ranking performance. As detailed in Appendix C.1, there is a connection between correct anomaly score ranking and high AUROC performance, a common evaluation metric for anomaly detection.

Existing methods use querying strategies that do not have good coverage and are therefore not optimal under Theorem 4.1. For a limited querying budget, random querying puts too much weight on high-density areas of the data space, while other strategies only query locally, e.g., close to an estimated decision boundary between normal and abnormal data.

Proposed Querying Strategy. Based on Theorem 4.1, we propose a querying strategy that encourages tight coverage: diverse querying. In practice, we use the seeding algorithm of

k-means++ which is usually used to initialize diverse clusters.[1] It iteratively samples another data point to be added to the query set $\mathcal{Q}$ until the labeling budget is reached. Given the existing queried samples, the probability of drawing another query from the unlabeled set $\mathcal{U}$ is proportional to its distance to the closest sample already in the query set $\mathcal{Q}$:

$$p_{\text{query}}(\mathbf{x}_i) = \text{softmax}\big(h(\mathbf{x}_i)/\tau\big) \quad \forall i \in \mathcal{U}, \tag{4.2}$$

The temperature parameter τ controls the diversity of the sampling procedure, and $h(\mathbf{x}_i) = \min_{\mathbf{x}_j \in \mathcal{Q}} d(\mathbf{x}_i, \mathbf{x}_j)$ is the distance of a sample $\mathbf{x}_i$ to the query set $\mathcal{Q}$. For a meaningful notion of distance, we define d in an embedding space as $d(\mathbf{x}, \mathbf{x}') = \|\phi(\mathbf{x}) - \phi(\mathbf{x}')\|_2$, where ϕ is a neural feature map. We stress that all deep methods considered in this paper have an associated feature map that we can use. The fact that L2 distance is used in the querying strategy is not an ad-hoc choice but rather aligned with the δ-ball radius definition (Equation (C.1) in Appendix C.1) in Theorem 4.1.

In Appendix C.1, we discuss the cover radius and empirically validate that diverse querying leads to smaller δ than others and is hence advantageous for anomaly detection.

4.3.5 Semi-Supervised Outlier Exposure Loss

We next consider how to use both labeled and unlabeled samples in training. We propose SOEL whose loss combines the unsupervised anomaly detection loss of LOE [Qiu et al., 2022b] for the unlabeled data with the supervised loss (Equation (4.1)) for the labeled samples. For all queried data (with index set $\mathcal{Q}$), we assume that ground truth labels y_i are available, while for unqueried data (with index set $\mathcal{U}$), the labels $\tilde{y}_i$ are unknown. Adding

[1] This has complexity $O(KN)$ which can be reduced to $O(K \log N)$ using scalable alternatives [Bahmani et al., 2012].

both losses together yields

$$\mathcal{L}(\theta, \tilde{\mathbf{y}}) = \frac{1}{|\mathcal{Q}|} \sum_{j \in \mathcal{Q}} \big(y_j \mathcal{L}_a^\theta(\mathbf{x}_j) + (1 - y_j) \mathcal{L}_n^\theta(\mathbf{x}_j) \big) + \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \big(\tilde{y}_i \mathcal{L}_a^\theta(\mathbf{x}_i) + (1 - \tilde{y}_i) \mathcal{L}_n^\theta(\mathbf{x}_i) \big). \tag{4.3}$$

Similar to Qiu et al. [2022b], optimizing this loss involves a block coordinate ascent scheme that alternates between inferring the unknown labels and taking gradient steps to minimize Equation (4.3) with the inferred labels. In each iteration, the pseudo labels $\tilde{y}_i$ for $i \in \mathcal{U}$ are obtained by minimizing Equation (4.3) subject to a constraint of $\sum_{i \in \mathcal{Q}} y_i + \sum_{i \in \mathcal{U}} \tilde{y}_i = \alpha N$. The constraint ensures that the inferred anomaly labels respect a certain contamination ratio α. To be specific, let $\tilde{\alpha}$ denote the fraction of anomalies among the *unqueried* set $\mathcal{U}$, so that $\tilde{\alpha}|\mathcal{U}| + \sum_{j \in \mathcal{Q}} y_j = \alpha N$. The constrained optimization problem is then solved by using the current anomaly score function S to rank the unlabeled samples and assign the top $\tilde{\alpha}$-quantile of the associated labels $\tilde{y}_i$ to the value 1, and the remaining to the value 0.

We illustrate SOEL's effect on a 2D toy data example in Figure 4.1, where SOEL (**d**) almost achieves the same performance as the supervised AD (**c**) with only one queried point.

In theory, α could be treated as a hyperparameter, 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 Qiu et al. [2022b] have to assume that the contamination ratio is given, SOEL provides an opportunity to estimate α. In Section 4.3.6, we develop an importance-sampling based approach to estimate α from the labeled data. Estimating this ratio can be beneficial for many anomaly detection algorithms, including OC-SVM [Schölkopf et al., 2001], kNN [Ramaswamy et al., 2000], Robust PCA/Auto-encoder [Zhou and Paffenroth, 2017], and Soft-boundary deep SVDD [Ruff et al., 2018]. When working with contaminated data, these algorithms require a decent estimate of the contamination ratio for good performance.

Another noteworthy aspect of the SOEL loss is that it weighs the *averaged* losses equally to each other. In Appendix C.5.9, we empirically show that equal weighting yields the best results among a large range of various weights. This provides more weight to every queried data point than to an unqueried one, because we expect the labeled samples to be more informative. On the other hand, it ensures that neither loss component will dominate the learning task. Our equal weighting scheme is also practical because it avoids a hyperparameter.

4.3.6 Contamination Ratio Estimation.

To eliminate a critical hyperparameter in our approach, we estimate the *contamination ratio* α, i.e., the fraction of anomalies in the dataset. Under a few assumptions, we show how to estimate this parameter using mini-batches composed of on non-i.i.d. samples.

We consider the contamination ratio α as the fraction of anomalies in the data. We draw on the notation from Section 4.3.1 to define $y(\mathbf{x})$ as an oracle, outputting 1 if $\mathbf{x}$ is an anomaly, and 0 otherwise (e.g., upon *querying* $\mathbf{x}$). We can now write $\alpha = \mathbb{E}_{p(\mathbf{x})}[y(\mathbf{x})]$.

Estimating α would be trivial given an unlimited querying budget of i.i.d. data samples. The difficulty arises due to the fact that (1) our querying budget is limited, and (2) we query data in a non-i.i.d. fashion so that the sample average is not representative of the anomaly ratio of the full data set.

Since the queried data points are not independently sampled, we cannot straightforwardly estimate α based on the empirical frequency of anomalies in the query $\mathcal{Q}$. More precisely, our querying procedure results in a chain of indices $\mathcal{Q} = \{i_1, i_2, ..., i_{|\mathcal{Q}|}\}$, where $i_1 \sim \text{Unif}(1 : \text{N})$, and each conditional distribution $i_k | i_{<k}$ is defined by Equation (4.2). We will show as follows that this sampling bias can be compensated using importance weights.

As follows, we first propose an importance-weighted estimator of α and then prove the

estimator is unbiased under certain idealized conditions specified by two assumptions about our querying strategy. Justifications for the two assumptions will be provided below.

For a random query $\mathcal{Q}$, its anomaly scores $\{S(\mathbf{x}_i) : i \in \mathcal{Q}\}$ and anomaly labels $\{y(\mathbf{x}_i) : i \in \mathcal{Q}\}$ are known. Write $S(\mathbf{x}_i)$ as s_i and let $p_s(s_i)$ denote the marginal density of population anomaly scores and $q_s(s_i)$ denote the marginal density of the queried samples' anomaly scores. Our importance-weighted estimator of the contamination ratio is

$$\hat{\alpha} = \frac{1}{|\mathcal{Q}|} \sum_{i=1}^{|\mathcal{Q}|} \frac{p_s(s_i)}{q_s(s_i)} y(\mathbf{x}_i). \tag{4.4}$$

As discussed above, $y(\mathbf{x}_i)$ are the ground truth anomaly labels, obtained from querying $\mathcal{Q}$. The estimator takes into account that, upon repulsive sampling, we will sample data points in the tail regions of the data distribution more often than we would upon uniform sampling.

In practice, we learn p_s and q_s using a kernel density estimator in the one-dimensional space of anomaly scores of the training data and the queried data, respectively. We set the bandwidth to the average spacing of scores. With the following two assumptions, Equation (4.4) is unbiased.

Assumption 1. *The anomaly scores $\{S(\mathbf{x}_i) : i \in \mathcal{Q}\}$ in a query set $\mathcal{Q}$ are approximately independently distributed.*

Assumption 2. *Let $y_s(S(\mathbf{x}))$ denote an oracle that assigns ground truth anomaly labels based on the model's anomaly scores $S(\mathbf{x})$. We assume that such an oracle exists, i.e., the anomaly score $S(\mathbf{x})$ is a sufficient statistic of the ground truth anomaly labeling function: $y_s(S(\mathbf{x})) = y(\mathbf{x})$.*

Assumptions 1 and 2 are only approximations of reality. In our experiment section, we will show that they are good working assumptions to estimate anomaly ratios. Below, we will provide additional strong evidence that assumptions 1 and 2 are well justified.

The following theorem is a consequence of them:

Theorem 4.2. *Assume that Assumptions 1 and 2 hold. Then, Equation* (4.4) *is an unbiased estimator of the contamination ratio* α*, i.e.,* $\mathbb{E}[\hat{\alpha}] = \alpha$*.*

The proof is in Appendix C.2. Theorem 4.2 allows us to estimate the contamination ratio based on a non-iid query set $\mathcal{Q}$.

Discussion. We empirically verified the fact that Theorem 4.2 results in reliable estimates for varying contamination ratios in Appendix C.2.4. Since Assumptions 1 and 2 seem strong, we discuss their justifications and empirical validity next.

While verifying the independence assumption (Assumption 1) rigorously is difficult, we tested for linear correlations between the scores (Appendix C.2.2). We found that the absolute off-diagonal coefficient values are significantly smaller than one on CIFAR-10, providing support for Assumption 1. A heuristic argument can be provided to support the validity of Assumption 1 based on the following intuition. When data points are sampled diversely in a high-dimensional space, the negative correlations induced by their repulsive nature tend to diminish when the data is projected onto a one-dimensional subspace. This intuition stems from the fact that a high-dimensional ambient space offers ample dimensions for the data points to avoid clustering. To illustrate this, consider the scenario of sampling diverse locations on the Earth's surface, with each location representing a point in the high-dimensional space. By including points from various continents, we ensure diversity in their spatial distribution. However, when focusing solely on the altitude of these locations (such as distinguishing between mountain tops and flat land), it is plausible that the altitude levels are completely uncorrelated. While this heuristic argument provides an intuitive understanding, it is important to note that it does not offer a rigorous mathematical proof.

To test Assumption 2, we tested the degree to which the anomaly score is a sufficient statistic for anomaly scoring on the training set. The assumption would be violated if we could find

Table 4.1: A summary of all compared experimental methods' query strategy and training strategy irrespective of their backbone models.

Name	Reference	Querying Strategy	Loss (labeled)	Loss (unlabeled)
Mar	Görnitz et al. [2013]	margin query	superv. (Equation (4.1))	one class
Hybr1	Görnitz et al. [2013]	margin diverse query	superv. (Equation (4.1))	one class
Pos1	Pimentel et al. [2020]	most positive query	superv. (Equation (4.1))	none
Pos2	Barnabé-Lortie et al. [2015]	most positive query	superv. (Equation (4.1))	one class
Rand1	Ruff et al. [2019]	random query	superv. (Equation (4.1))	one class
Rand2	Trittenbach et al. [2021]	positive random query	superv. (Equation (4.1))	one class
Hybr2	Das et al. [2019]	positive diverse query	superv. (Equation (4.1))	none
Hybr3	Ning et al. [2022]	positive diverse query	refinement	weighted one class
SOEL	[ours]	diverse (Equation (4.2))	semi-supervised outlier exposure loss (Equation (4.3))	

pairs of training data $\mathbf{x}_i$ and $\mathbf{x}_j$, where $\mathbf{x}_i \neq \mathbf{x}_j$, with identical anomaly scores $S(\mathbf{x}_i) = S(\mathbf{x}_j)$ but different anomaly labels $y_s(s_i) \neq y_s(s_j)$[2]. On FMNIST, we found 38 data pairs with matching scores, and none of them had opposite anomaly labels. For CIFAR-10, the numbers were 21 and 3, respectively. See Appendix C.2.3 for details.

4.4 Experiments

We study SOEL on standard image benchmarks, medical images, tabular data, and surveillance videos. Our extensive empirical study establishes how our proposed method compares to eight anomaly detection methods with labeling budgets implemented as baselines. We first describe the baselines and their implementations (Table 4.1) and then the experiments on images (Section 4.4.1), tabular data (Section 4.4.2), videos (Section 4.4.3) and finally additional experiments (Section 4.4.4).

Baselines. Most existing baselines apply their proposed querying and training strategies to shallow anomaly detection methods or sub-optimal deep models (e.g., autoencoders [Zhou and Paffenroth, 2017]). In recent years, these approaches have consistently been outperformed by self-supervised anomaly detection methods [Hendrycks et al., 2019]. For a fair

[2]The condition $S(\mathbf{x}_i) \neq S(\mathbf{x}_j)$ for $\mathbf{x}_i \neq \mathbf{x}_j$ hints we can assign a unique label to each data point based on their scores.

Table 4.2: AUC (%) with standard deviation for anomaly detection on 11 image datasets when the query budget $|\mathcal{Q}| = 20$. SOEL outperforms all baselines by a large margin by querying diverse and informative samples.

	Mar	**Hybr1**	**Pos1**	**Pos2**	**Rand1**	**Rand2**	**Hybr2**	**Hybr3**	**SOEL**
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**
Pneumonia	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**

comparison, we endow all baselines with the same self-supervised backbone models also used in our method. By default we use NTL [Qiu et al., 2021] as the backbone model, which was identified as state-of-the-art in a recent independent comparison of 13 models [Alvarez et al., 2022]. Results with other backbone models are shown in Appendix C.5.2.

The baselines are summarized in Table 4.1 and detailed in Appendix C.3. They differ in their querying strategies (col. 3) and training strategies (col. 4 & 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 (4.1)). As an exception, Ning et al. [2022] remove all queried anomalies and then train a weighted one-class objective on the remaining data. All baselines weigh the unsupervised and supervised losses equally. They differ in their querying strategies, summarized below:

- **Margin query** selects samples close to the boundary of the normality region deterministically. The method uses the true contamination ratio to choose an ideal boundary.
- **Margin diverse query** combines margin query with neighborhood-based diversification. It selects samples that are not k-nearest neighbors of the queried set. Thus samples are both diverse and close to the boundary.
- **Most positive query** always selects the top-ranked samples ordered by their anomaly

scores.

- **Positive diverse query** combines querying according to anomaly scores with distance-based diversification. The selection criterion combines anomaly score and the minimum Euclidean distance to all queried samples.
- **Random query** draws samples uniformly.
- **Positive random query** samples uniformly among the top 50% data ranked by anomaly scores.

Implementation Details. In all experiments, we use a NTL [Qiu et al., 2021] backbone model for all methods. Experiments with other backbone models are shown in Appendix C.5.2. On images and videos, NTL is built upon the penultimate layer output of a frozen ResNet-152 pre-trained on ImageNet. NTL is trained for one epoch, after which all $|\mathcal{Q}|$ queries are labeled at once. The contamination ratio α in SOEL is estimated immediately after the querying step and then fixed for the remaining training process. We follow Qiu et al. [2022b] and set $\tilde{y}_i = 0.5$ for inferred anomalies. This accounts for the uncertainty of whether the sample truly is an anomaly. More details are given in Appendix C.4 and Algorithm 2.

4.4.1 Experiments on Image Data

We study SOEL on standard image benchmarks to establish how it compares to eight well-known baselines with various querying and training strategies. Informative querying plays an important role in medical domains where expert labeling is expensive. Hence, we also study nine medical datasets from Yang et al. [2021b]. We describe the datasets, the evaluation protocol, and finally the results of our study.

Image Benchmarks. We experiment with two popular image benchmarks: CIFAR-10 and Fashion-MNIST. These have been widely used in previous papers on deep anomaly detection

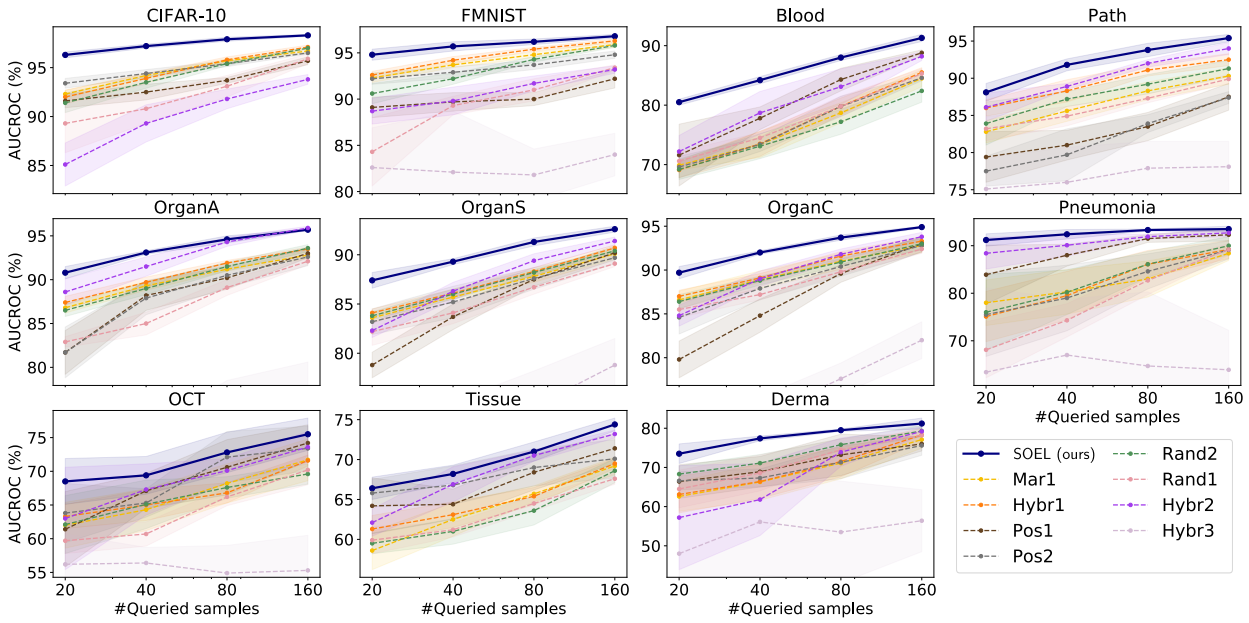


Figure 4.2: Running AUCs (%) with different query budgets. Models are evaluated at $20, 40, 80, 160$ queries. SOEL performs the best among the compared methods on all query budgets.

[Ruff et al., 2018, Golan and El-Yaniv, 2018, Hendrycks et al., 2019, Bergman and Hoshen, 2020].

Medical Images. Since medical imaging is an important practical application of anomaly detection, we also study SOEL 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. [2021b] that have more than 500 samples in each class, including Blood, OrganA, OrganC, OrganS, OCT, Pathology, Pneumonia, and Tissue. We also include Dermatoscope but restricted to the classes with more than 500 training samples.

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 [Ruff et al., 2018, Hendrycks et al., 2019, Bergman and Hoshen, 2020]. While respecting the original train and test split of these datasets, the protocol iterates over the classes and treats

Table 4.3: F1-score (%) with standard deviation for anomaly detection on tabular data when the query budget $|\mathcal{Q}| = 10$. SOEL performs the best on 3 of 4 datasets and outperforms all baselines by 3.2 percentage points on average.

	Mar	**Hybr1**	**Pos1**	**Pos2**	**Rand1**	**Rand2**	**Hybr2**	**Hybr3**	**SOEL**
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**
Ionosphere	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**

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% [Ruff et al., 2019, Wang et al., 2019, Qiu et al., 2022b]. This protocol can simulate a "human expert" to provide labels for the queried samples because the datasets provide ground-truth class labels. The reported results (in terms of AUC %) for each dataset are averaged over the number of experiments (i.e., classes) and over five independent runs.

Results. We report the evaluation results of our method (SOEL) and the eight baselines on all eleven image datasets in Table 4.2. When querying 20 samples, our proposed method SOEL significantly outperforms the best-performing baseline by 6 percentage points on average across all datasets. We also study detection performance as the query budget increases from 20 to 160 in Figure 4.2. The results show that, with a small budget of 20 samples, SOEL (by querying diverse and informative samples) makes better usage of the labels than the other baselines and thus leads to better performance by a large margin. As more samples are queried, the performance of almost all methods increases but even for 160 queries when the added benefit from adding more queries starts to saturate, SOEL still outperforms the baselines.

4.4.2 Experiments on Tabular Data

Many practical use cases of anomaly detection (e.g., in health care or cyber security) are concerned with tabular data. For this reason, we study SOEL on four tabular datasets from various domains. We find that it outperforms existing baselines, even with as few as 10 queries. We also confirmed the fact that our deep models are competitive with classical methods for tabular data in Appendix C.5.13.

Tabular Datasets. Our study includes the four multi-dimensional tabular datasets from the ODDS repository 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 (which is randomly sub-sampled to half its size) to achieve a contamination ratio of 10%. The datasets are BreastW, Ionosphere, Pima, and Satellite. As in the image experiments, there is one round of querying, in which 10 samples are labeled. For each dataset, we report the averaged F1-score (%) with standard deviations over five runs with random train-test splits and random initialization.

Results. SOEL performs best on 3 of 4 datasets and outperforms all baselines by 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 on tabular data.

4.4.3 Experiments on Video Data

Detecting unusual objects in surveillance videos is an important application area for anomaly detection. Due to the large variability in abnormal objects and suspicious behavior in surveillance videos, expert feedback is very valuable to train an anomaly detector in a semi-supervised manner. We use NTL as the backbone model and study SOEL on a public

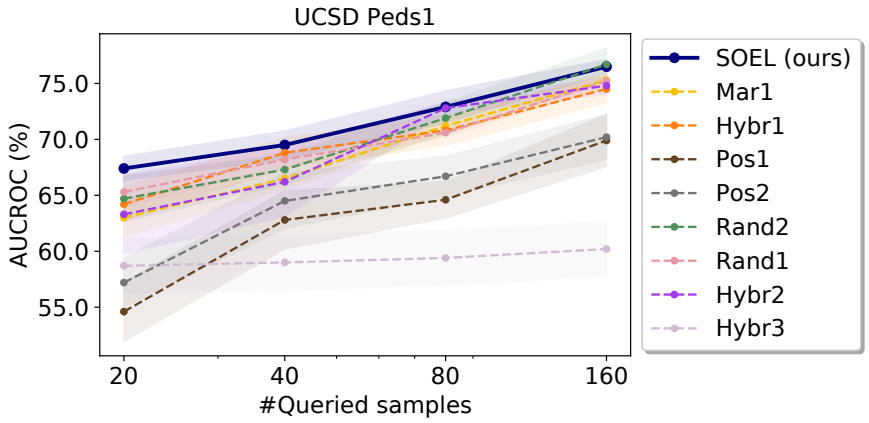


Figure 4.3: Results on the video dataset UCSD Peds1 with different query budgets. SOEL achieves the leading performance.

surveillance video dataset (UCSD Peds1). The goal is to detect abnormal video frames that contain non-pedestrian objects.

Following Pang et al. [2020], we subsample a mix of normal and abnormal frames for training (using an anomaly ratio of 0.3) and use the remaining frames for testing. Before running any of the methods, a ResNet pretrained on ImageNet is used to obtain a fixed feature vector for each frame. We vary the query budget from $|\mathcal{Q}| = 20$ to $|\mathcal{Q}| = 160$ and compare SOEL to all baselines. Results in terms of average AUC and standard error over five independent runs are reported in Figure 4.3. SOEL consistently outperforms all baselines, especially for smaller querying budgets.

4.4.4 Additional Experiments

In Appendix C.5, we provide additional experiments and ablations demonstrating SOEL's strong performance and justifying modeling choices. The three most important findings are:

- **SOEL vs. Active Learning:** Our framework is superior to its extension to the sequential active learning (Figure C.6).

- **Varying Contamination Ratio:** Figure C.4 demonstrates that SOEL dominates under varying contamination ratios (1%, 5%, 20%). In addition, Table C.1 confirms that Equation (4.4) reliably estimates α on both CIFAR-10 and F-MNIST.
- **Backbone Models:** Table C.3 shows that SOEL also performs best for the backbone models MHRot [Hendrycks et al., 2019] and DSVDD [Ruff et al., 2018].

In addition, we provide an ablation of the temperature τ (Table C.5), a discussion on the effects of initialization randomness (Appendix C.5.1), an ablation study of the pseudo-label $\tilde{y}$ values (Table C.6), a comparison to binary classification (Figure C.5), an ablation of the SOEL loss components, an ablation of querying strategies (Figure C.7), additional methods for inferring y (Figure C.10), and comparison to additional semi-supervised baselines (Figure C.10, Table C.7).

4.5 Conclusion

We introduced semi-supervised outlier exposure with a limited labeling budget (SOEL). Inspired by a set of conditions that guarantee the generalization of anomaly score rankings from queried to unqueried data, we proposed to use a diversified querying strategy and a combination of two losses for queried and unqueried samples. By weighting the losses equally 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, making it easy to use. An extensive empirical study on images, tabular data, and video confirmed the efficacy of SOEL as a semi-supervised learning framework compatible with many existing losses for anomaly detection.

Limitations: The success of our approach relies on several heuristics that we demonstrated were empirically effective but that cannot be proven rigorously. Estimation of the contamination ratio can be noisy when the query set is small—but the LOE loss is robust even

under misspecification of the contamination ratio [Qiu et al., 2022b]. The diversified sampling strategy becomes expensive when the dataset is large, but this can be mitigated by random data thinning.

Societal Impacts: The use of human labels for anomaly detection runs the risk of introducing potential human biases in the definition of what is anomalous, particularly for datasets involving human subjects. Since our approach relies heavily on a relatively small number of human labels, the deployment of our approach with real human labelers would benefit by having guidelines for the labelers in terms of providing fair labels and avoiding amplification of bias.

Chapter 5

Zero-Shot Anomaly Detection via Batch Normalization

This chapter is based on a published paper at NeurIPS 2023: *Zero-Shot Anomaly Detection via Batch Normalization* by Aodong Li*, Chen Qiu*, Marius Kloft, Padhraic Smyth, Maja Rudolph, Stephan Mandt [Li et al., 2024]

5.1 Introduction

Anomaly detection—the task of identifying data instances deviating from the norm [Ruff et al., 2021]—plays a significant role in numerous application domains, such as fake review identification, bot detection in social networks, tumor recognition, and industrial fault detection. AD is particularly crucial in safety-critical applications where failing to recognize anomalies, for example, in a chemical plant or a self-driving car, can risk lives.

Consider a medical setting where an anomaly detector encounters a batch of medical images from different patients. The medical images have been recorded with a new imaging

technology different from the training data, or the patients are from a demographic the anomaly detector has not been trained on. Our goal is to develop an anomaly detector that can still process such data using batches, assigning low scores to normal images and high scores to anomalies (i.e., images that differ systematically) without retraining. To achieve this zero-shot adaptation, we exploit the fact that anomalies are rare. Given a new batch of test data, a zero-shot anomaly detection method [Liznerski et al., 2022, Esmaeilpour et al., 2022, Schwartz et al., 2022, Jeong et al., 2023] has to detect which features are typical of the majority of normal samples and which features are atypical.

We propose ACR, a lightweight zero-shot anomaly detection method that combines two simple ideas: batch normalization and meta-training. Assuming an overall majority of "normal" samples, a randomly-sampled batch will typically have more normal samples than anomalies. The effect of batch normalization is then to draw these normal samples closer to the center (in its recentering and scaling operation), while anomalies will end up further away from the center. Notably, this scaling and centering is robust to a distribution shift in the input, allowing a self-supervised anomaly detector to generalize to distributions never encountered during training. We propose a meta-training scheme to unlock the power of batch normalization layers for zero-shot anomaly detection. During training, the anomaly detector will see many different anomaly detection tasks, mixed from different choices for normal and abnormal examples. Through this variability in the training tasks, the anomaly detector will learn to rely as much as possible on the batch normalization operations in its architecture.

Advantages of ACR include that it is theoretically grounded, simple, domain-independent, and compatible with various backbone models commonly used in deep anomaly detection [Ruff et al., 2018, Qiu et al., 2021]. Contrary to recent approaches based on foundation models [Jeong et al., 2023], applicable only to images, ACR can be employed on data from any domain, such as time series, tabular data, or graphs.

We begin by presenting our assumptions and method in Section 5.2. Next, with the main

idea in mind, we describe the related work in Section 5.3. We demonstrate the effectiveness of our method with experiments in Section 5.4. Finally, we conclude our work and state the limitations and societal impacts.

Our contributions can be summarized as follows:

- **An effective new method.** Our results for the first time show that training off-the-shelf deep anomaly detectors on a meta-training set, using batch normalization layers, gives automatic zero-shot generalization for anomaly detection. For which we derive a generalization bound on anomaly scores.
- **Zero-shot anomaly detection on tabular data.** We provide the first empirical study of zero-shot anomaly detection on tabular data, where our adaptation approach retains high accuracy.
- **Competitive results for images.** Our results demonstrate not only a substantial improvement in zero-shot anomaly detection performance for non-natural images, including medical imaging but also establish a new state-of-the-art in anomaly segmentation on the MVTec AD benchmark [Bergmann et al., 2019].

5.2 Method

We begin with the problem statement in Section 5.2.1 and then state the assumptions in Section 5.2.2. Finally we present our proposed solution in Section 5.2.3. The training procedure is outlined in Algorithm 3 in Appendix D.3.

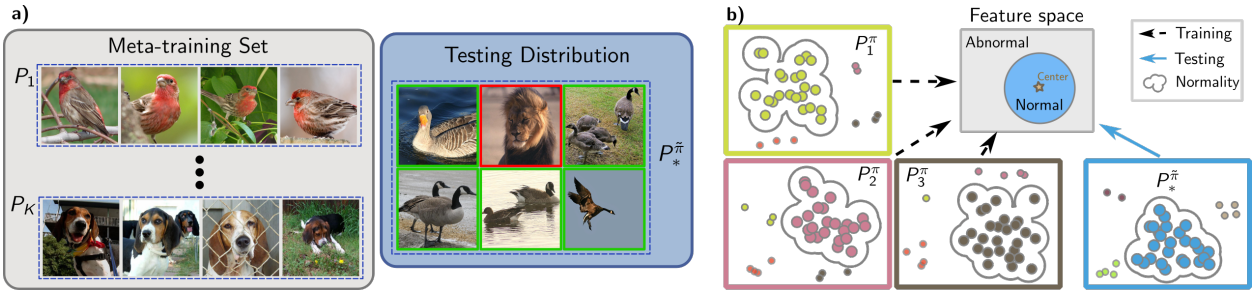


Figure 5.1: **a)** Demonstrations of concrete examples of a meta-training set and a testing distribution. It is not necessary for the meta-training set to include the exact types of samples encountered during testing. For instance, when detecting lions within geese, the training data does not need to include lions or geese. **b)** Illustration of zero-shot batch-level anomaly detection with ACR using a one-class classifier [Ruff et al., 2018]. The approach encounters three tasks ($P^{\pi}_{1:3}$, Equation (5.6)) during training (black arrows) and learns to map each task's majority of samples (i.e., the normal samples) to a shared learned center in embedding space. At test time (blue arrow), the learned model maps the normal (majority) samples to the same center and the distance from the center serves as anomaly detection score.

5.2.1 Problem Statement and Method Overview

We consider the problem of learning an anomaly detector that is required to immediately adapt (without any further training) when deployed in a new environment. The main idea is to use batch normalization as a mechanism for *adaptive batch-level* anomaly detection. For any batch of data containing mostly "normal" samples, each batch normalization shifts its inputs to the origin, thereby (1) enabling the discrimination between normal data and outliers/anomalies, and (2) bringing data from different distributions into a common frame of reference. (Notably, we propose applying batch norm in multiple layers for different anomaly scorers.) For the algorithm to generalize to unseen distributions, we train our model on multiple data sets of "normal" data simultaneously, making sure each training batch contains a majority of related data points (from the same distribution) at a time.

Figure 5.1a illustrates this idea, where all distributions are exemplified based on the example of homogeneous groups of animals (only dogs, only robins, etc.) The goal is to detect a lion among geese, where neither geese nor lions have been encountered before. Figure 5.1b

illustrates the scheme based on the popular example of deep support vector data description (DeepSVDD) [Ruff et al., 2018], where samples are mapped to a pre-specified point in an embedding space and scored based on their distance to this point. All training distributions are mapped to the same point, as enabled through batch normalization.

5.2.2 Notation and Assumptions

To formalize the notion of a meta-training set, we consider a distribution of interrelated data distributions (previously referred to as groups) as commonly studied in meta-learning and zero-shot learning [Baxter, 2000, Finn et al., 2017, Frikha et al., 2021, Huang et al., 2022]. This inter-relatedness can be expressed by assuming that K training distributions $P_1, \ldots, P_K$ and a test distribution P_* are sampled from a meta-distribution $\mathcal{P}$:

$$P_1, \cdots, P_K, P_* \overset{\text{i.i.d.}}{\sim} \mathcal{P}. \tag{5.1}$$

We assume that the distributions in $\mathcal{P}$ share some common structure, such that training a model on one distribution has the potential to aid in deploying the model on another distribution. For example, the data $\mathbf{x}$ could be radiology images from patients, and each P_j or P_* could be a distribution of images from a specific hospital. These distributions share similarities but differ systematically because of differences in radiology equipment, calibration, and patient demographics[1]. Each of the distributions $P \in \mathcal{P}$ defines a different anomaly detection task. For each task, we have to obtain an anomaly scoring function $S(\mathbf{x}; \theta)$ that assigns low scores to normal samples $\mathbf{x} \sim P$ and high scores to anomalies.

We now consider a batch $\mathcal{B} \subset \mathcal{D}$ of size B, taken from an underlying data set $\mathcal{D} \sim P$ of size

[1]The divergence between distributions can be much larger than shown in this example. See our experiments.

N. The batch can be characterized by indexing data points from $\mathcal{D}$:

$$\mathcal{B} \equiv (i_1, ..., i_B) \sim \text{Unif}(\{1, ..., N\}). \tag{5.2}$$

We denote the anomaly scores on a *batch* level by defining a vector-valued anomaly score

$$\mathbf{S}(\mathbf{x}_\mathcal{B}; \theta) = \big(\mathbf{S}_{i_1}(\mathbf{x}_\mathcal{B}; \theta), \cdots, \mathbf{S}_{i_B}(\mathbf{x}_\mathcal{B}; \theta)\big), \tag{5.3}$$

indicating the anomaly score for every datum in a batch. By thresholding the anomaly scores $\mathbf{S}_i(\mathbf{x}_\mathcal{B}; \theta)$, we obtain binary predictions of whether data point $\mathbf{x}_i$ is anomalous in *the context of batch* $\mathbf{x}_\mathcal{B}$.

By conditioning on a batch of samples, our approach obtains distributional information beyond a single sample. For example, an image of a cat may be normal in the context of a batch of cat images, but it may be anomalous in the context of a batch of otherwise dog images. This is different from current deep anomaly detection schemes that evaluate anomaly scores without referring to a context.

Before presenting a learning scheme of how to combine batch-level information in conjunction with established anomaly detection approaches, we discuss the assumptions that our approach makes. (The empirical or theoretical justifications as well as possibilities of removing or mitigating the assumptions can be found in Appendix D.1.)

A1 *Availability of a meta-training set.* As discussed above, we assume the availability of a set of interrelated distributions. The meta-set is used to learn a model that can adapt without re-training.

A2 *Batch-level anomaly detection.* As mentioned above, we assume we perform batch-level predictions at test time, allowing us to detect anomalies based on reference data in the batch.

A3 *Majority of normal data.* We assume that normal data points take the majority in every i.i.d. sampled test batch.

Due to the absence of anomaly labels (or text descriptions) at test-time, we cannot infer the correct anomaly labels without assumptions **A2** and **A3**. Together, they instruct us that given a batch of test examples, the majority of the samples in the batch constitute normal samples.

5.2.3 Adaptively Centered Representations

Batch Normalization as Adaptation Modules. An important component of our method is batch normalization, which shifts and re-scales any data batch $\mathbf{x}_{\mathcal{B}}$ to have a sample mean zero and variance one. Batch normalization also provides a naïve parameter-free zero-shot batch-level anomaly detector:

$$\mathbf{S}_i^{\text{naïve}}(\mathbf{x}_{\mathcal{B}}) = ||(\mathbf{x}_i - \bar{\mu}_{\mathbf{x}_{\mathcal{B}}})/\bar{\sigma}_{\mathbf{x}_{\mathcal{B}}}||_2^2, \tag{5.4}$$

where $\bar{\mu}$ and $\bar{\sigma}^2$ are the coordinate-wise sample mean and sample variance. $\bar{\mu}$ is dominated by the majority of the batch, which by assumption A3, is the normal data. If the $\mathbf{x}_i$ lie in an informative feature space, anomalies will have a higher-than-usual distance to the mean, making the approach a simple, *adaptive* anomaly detection method, illustrated in Figure D.1 in Appendix D.4.

While the example provides a proof of concept, in practice, the normal samples typically do not concentrate around their mean in the raw data space. Next, we integrate this idea into neural networks and develop an approach that learns adaptively centered representations for zero-shot anomaly detection.

Deep Neural Networks with Batch Normalization Layers as Scalable Zero-shot Anomaly Detectors. In deep neural networks, the adaptation ability is obtained for *free* with batch normalization layers [Ioffe and Szegedy, 2015]. Batch normalization has become a standard and necessary component to facilitate optimization convergence in training neural networks. In common neural network architectures [He et al., 2016, Huang et al., 2017, Radosavovic et al., 2020], batch normalization layers are used after each non-linear transformation layer, making a zero-shot adaptation with respect to its input batch. The entire neural network, stacking up many non-linear transformation and normalization layers, has powerful potential in scalable zero-shot adaptation and learning adaptation-needed feature representations for complex data forms.

Training Objective. As discussed above, we can instantiate $S(\mathbf{x};\theta)$ as a deep neural network with batch normalization layers and optimize the neural network weights θ. We first provide our objective function and then the rationality. Our approach is compatible with a wide range of deep anomaly detection objectives; therefore we consider a generic loss function $\mathbf{L}_n^\theta(\mathbf{x}_\mathcal{B}) := \mathcal{L}_n[\mathbf{S}(\mathbf{x}_\mathcal{B};\theta)] = \big(\mathcal{L}_n[\mathbf{S}_{i_i}(\mathbf{x}_\mathcal{B};\theta)], \cdots, \mathcal{L}_n[\mathbf{S}_{i_B}(\mathbf{x}_\mathcal{B};\theta)]\big)$ that is a function of the anomaly score[2]. For example, in many cases, the loss function to be minimized is the anomaly score itself (averaged over the batch). We denote the loss function on each data point in the batch by $\mathbf{L}_{n,i}^\theta(\mathbf{x}_\mathcal{B}) := \mathcal{L}_n[\mathbf{S}_i(\mathbf{x}_\mathcal{B};\theta)]$ for simplicity.

The availability of a meta-data set (**A1**) gives rise to the following minimization problem:

$$\theta^* = \arg\min_\theta \tfrac{1}{K}\textstyle\sum_{j=1}^K \mathbb{E}_{\mathbf{x}_\mathcal{B}\sim P_j}\mathbf{L}_n^\theta(\mathbf{x}_\mathcal{B}). \tag{5.5}$$

Typical choices for $\mathbf{L}_n^\theta(\mathbf{x}_\mathcal{B})$ include DeepSVDD [Ruff et al., 2019] and NTL [Qiu et al., 2021]. Details and modifications of this objective will follow.

[2]The subscript n suggests the loss function acts on normal data.

Comparison against Stationary Anomaly Detection.

Zero-shot AD (ours)	**Stationary AD**
▷ Batch-mode training	▷ Batch-mode training
BN layers are active	BN layers are active
$\theta^*_{\text{ACR}} = \arg\min_\theta \frac{1}{K}\sum_{j=1}^{K} \mathbb{E}_{\mathbf{x}_\mathcal{B}\sim P_j} \mathbf{L}^\theta(\mathbf{x}_\mathcal{B})$	$\theta^*_n = \arg\min_\theta \mathbb{E}_{\mathbf{x}_\mathcal{B}\sim P_n} \mathbf{L}^\theta(\mathbf{x}_\mathcal{B})$
▷ Batch-mode test	▷ Batch-mode test
BN layers are active	BN layers are off
$\mathbf{x}_\mathcal{B} \sim \pi P_* + (1-\pi)\bar{P}_*$	$\mathbf{x}_\mathcal{B} \sim \pi P_n + (1-\pi)\bar{P}_n$
$\mathbf{S}_{\theta^*_{\text{ACR}}}(\mathbf{x}_\mathcal{B}) = \{S^1_{\theta^*_{\text{ACR}}}(\mathbf{x}_\mathcal{B}), \dots, S^{\lvert\mathcal{B}\rvert}_{\theta^*_{\text{ACR}}}(\mathbf{x}_\mathcal{B})\}$	$\mathbf{S}_{\theta^*_n}(\mathbf{x}_\mathcal{B}) = \{S^1_{\theta^*_n}(\mathbf{x}_\mathcal{B}), \dots, S^{\lvert\mathcal{B}\rvert}_{\theta^*_n}(\mathbf{x}_\mathcal{B})\}$

Figure 5.2: Comparisons between the proposed zero-shot anomaly detection method and the regular anomaly detection approach where the normal data distribution is stationary. When training with mini-batches, both set the batch norm layers in the training mode. While stationary anomaly detection minimizes the loss function of a single normal data distribution, our method optimizes over K distributions. At test time, stationary anomaly detection sets the batch norm layers in inference mode, but our method still sets them in the training mode. Our approach allows the generalization of the test data from an unseen distribution P_* and its anomaly distribution $\bar{P}_*$.

Why does it work? Batch normalization helps re-calibrate the data batches of different distributions into similar forms: normal data will center around the origin. Such calibration happens from granular features (lower layers) to high-level features (higher layers), resulting in powerful feature learning and adaptation ability[3]. We visualize the calibration in Figure D.2 in Appendix D.9.2. Therefore, optimizing Equation (5.5) are able to learn a (locally) optimal $\mathbf{S}(\mathbf{x}; \theta^*)$ that is adaptive to all K *different* training distributions. Such learned adaptation ability will be guaranteed to generalize to unseen related distributions P_*. Figure 5.2 compares the proposed zero-shot anomaly detection framework against the regular anomaly detection framework at the training and testing time. See Section 5.2.4 below and Appendix D.2 for more details.

Meta Outlier Exposure. While Equation (5.5) can be a viable objective, we can significantly improve over it while avoiding trivial solutions[4]. The approach builds on treating

[3]Without batch normalization, optimizing Equation (5.5) can be meaningless for some objectives. See Appendix D.9.1.

[4]We explain how this objective avoids trivial solutions in Appendix D.7 and show the benefits in Table D.1 of Appendix D.9.1.

samples from other distributions as anomalies during training. The idea is that the synthetic anomalies can be used to guide learning a tighter decision boundary around the normal data [Hendrycks et al., 2018]. Drawing on the notation from Eq. 5.1, we thus simulate a mixture distribution by contaminating each P_j by admixing a fraction $(1-\pi) \ll 1$ of data from other available training distributions. The resulting corrupted distribution P_j^π is thereby

$$P_j^\pi := \pi P_j + (1-\pi)\bar{P}_j, \qquad \bar{P}_j := \tfrac{1}{K-1}\textstyle\sum_{i \neq j} P_i \tag{5.6}$$

This notation captures the case where the training distribution is free of anomalies ($\pi = 1$).

Next, we discuss constructing an additional loss for the admixed anomalies, whose identity is known at training time. As discussed in [Hendrycks et al., 2018, Qiu et al., 2022b], many loss functions $\mathbf{L}_n^\theta(\mathbf{x}_\mathcal{B})$ allow for easily constructing a loss $\mathbf{L}_a^\theta(\mathbf{x}_\mathcal{B})$ that behaves inversely. That means, we expect each item in $\mathbf{L}_a^\theta(\mathbf{x}_\mathcal{B})$ to be *large* when evaluated on normal samples, and small for anomalies. Importantly, both losses share the same parameters. In the context of DeepSVDD, we define $\mathbf{L}_n^\theta(\mathbf{x}_\mathcal{B}) = 1/\mathbf{L}_a^\theta(\mathbf{x}_\mathcal{B})$, but other definitions are possible for alternative losses [Ruff et al., 2018, 2019, Qiu et al., 2022b]. Using the inverse score, we can construct a supervised anomaly detection loss on the meta training set as follows.

We define a binary indicator variable y_i^j, indicating whether data point i is normal or anomalous in the context of distribution P_j (i.e., $y_i^j = 0$ iff $\mathbf{x}_{\mathcal{B},i} \in P_j$). We later refer to it as *anomaly label.* A natural replacement for the loss only on normal data $\mathbf{L}_n^\theta$ in Equation (5.5) is therefore

$$\mathbf{L}^\theta(\mathbf{x}_\mathcal{B}) = \frac{1}{B}\sum_{i \in \mathcal{B}} \{(1-y_i)\mathbf{L}_{n,i}^\theta(\mathbf{x}_\mathcal{B}) + y_i \mathbf{L}_{a,i}^\theta(\mathbf{x}_\mathcal{B})\}. \tag{5.7}$$

The loss function resembles the outlier exposure loss [Hendrycks et al., 2018], but as opposed to using synthetically generated samples (typically only available for images), we use samples

from the complement $\bar{P}_j$ at training time to synthesize outliers. The training pseudo-code is in Algorithm 3 of Appendix D.3.

In addition to DeepSVDD, we also study backbone models such as binary classifiers and NTL [Qiu et al., 2021]. For NTL, we adopt the $\mathbf{L}_n^\theta$ and $\mathbf{L}_a^\theta$ used by Qiu et al. [2022b]. For binary classifiers, we set $\mathbf{L}_n^\theta(\mathbf{x}) = -\log\big(1 - \sigma(f_\theta(\mathbf{x}))\big)$ and $\mathbf{L}_a^\theta(\mathbf{x}) = -\log\sigma(f_\theta(\mathbf{x}))$.

Batch-level Prediction. After training, we deploy the model in an unseen production environment to detect anomalies in a zero-shot adaptive fashion. Similar to the training set, the distribution will be a mixture of new normal samples P_* and an admixture of anomalies from a distribution never encountered before. For the method to work, we still assume that the majority of samples be normal (Assumption A3). Anomaly scores are assigned based on batches, as during training. For prediction, the anomaly scores are thresholded at a user-specified value.

Time complexity for prediction depends on the network complexity and is constant $O(1)$ relative to batch size, because the predictions can be trivially parallelized via modern deep learning libraries. We compare our zero-shot anomaly detection framework against the stationary anomaly detection approach.

5.2.4 Theoretical Results

Having described our method, we now establish a theoretical basis for ACR by deriving a bounded generalization error on an unseen test distribution P_*. We define the generalization error in terms of training and testing losses, i.e., we are interested in whether the expected loss generalizes from the meta-training distributions $P_1, \cdots, P_K$ to an unseen distribution P_*.

To prepare the notations, we split $\mathbf{L}^\theta(\mathbf{x})$ into two parts: a feature extractor $\mathbf{z} = f_\theta(\mathbf{x})$ that spans from the input layer to the last batch norm layer that performs batch normalization, and a loss function $\mathbf{L}(\mathbf{z})$ that covers all the remaining layers. When the input is only an individual data point, and there are no batch norm layers to shift the data, we write $\mathcal{L}(\mathbf{z})$ to differentiate the vector-valued loss. We use P_j^z to denote the data distribution P_j transformed by the feature extractor f_θ. We assume that P_j^z satisfies $\mathbb{E}_{\mathbf{z}\sim P_j^z}[\mathbf{z}] = 0$ and $\mathrm{Var}_{\mathbf{z}\sim P_j^z}[\mathbf{z}] = 1$ for $j = 1, \ldots, K, *$ because f_θ ends up with a batch norm layer.

Theorem 5.1. *Assume the mini-batches are large enough such that, for batches from each given distribution P_j, the mini-batch means and variances are approximately constant across batches. Furthermore, assume the loss function $\mathcal{L}(\mathbf{z})$ is bounded by C for any $\mathbf{z}$. Let $\|\cdot\|_{TV}$ denote the total variation. Then, the generalization error is upper bounded by*

$$\left|\mathbb{E}_{\mathbf{x}_\mathcal{B}\sim P_*}\left[\frac{1}{B}\sum_{i=1}^{B}\mathbf{L}_i^\theta(\mathbf{x}_\mathcal{B})\right] - \frac{1}{K}\sum_{j=1}^{K}\mathbb{E}_{\mathbf{x}_\mathcal{B}\sim P_j}\left[\frac{1}{B}\sum_{i=1}^{B}\mathbf{L}_i^\theta(\mathbf{x}_\mathcal{B})\right]\right| \leq C\left\|P_*^z - \frac{1}{K}\sum_{j=1}^{K}P_j^z\right\|_{TV}.$$

The proof is shown in Appendix D.2. Note that Theorem 5.1 still holds if P_j or P_* are contaminated distributions P_j^π or $P_*^{\tilde{\pi}}$.

Remark. Theorem 5.1 suggests that the generalization error of the expected loss function is bounded by the total variation distance between P_*^z and $\frac{1}{K}\sum_{j=1}^{K}P_j^z$. While we leave a formal bound of the TV distance to future studies, the following intuition holds: since f_θ contains batch norm layers, the empirical distributions $\frac{1}{K}\sum_{j=1}^{K}P_j^z$ and P_*^z will share the same (zero) mean and (unit) variance. If both distributions are dominated by their first two moments, we can expect the total variation distance to be small, providing an explanation for the approach's favorable generalization performance.

5.3 Related Work

Deep Anomaly Detection. Many recent advances in anomaly detection are built on deep learning methods [Ruff et al., 2021] and early strategies used autoencoder [Principi et al., 2017, Zhou and Paffenroth, 2017, Chen and Konukoglu, 2018] or density-based [Schlegl et al., 2017, Deecke et al., 2018] models. Another pioneering stream of research combined one-class classification [Schölkopf et al., 2001] with deep learning [Ruff et al., 2018, Qiu et al., 2022a]. Many other approaches to deep anomaly detection are self-supervised, employing a self-supervised loss function to train the detector and score anomalies [Golan and El-Yaniv, 2018, Hendrycks et al., 2019, Sohn et al., 2020b, Bergman and Hoshen, 2020, Qiu et al., 2021, Schneider et al., 2022, Shenkar and Wolf, 2021, Li et al., 2023].

All of these approaches assume that the data distribution will not change too much at test time. However, in many practical scenarios, there will be significant shifts in the abnormal distribution and even the normal distribution. For example, Dragoi et al. [2022] observed that existing anomaly detection methods fail in detecting anomalies when distribution shifts occur in network intrusion detection. Another line of work in this context requires test-time modeling for the entire test set, e.g., COPOD [Li et al., 2020], ECOD [Li et al., 2022], and robust autoencoder [Zhou and Paffenroth, 2017], preventing real-time deployment.

Few-shot Anomaly Detection. Several recent works have studied adapting an anomaly detector to shifts by fine-tuning a few test samples. One stream of research applies model-agnostic meta learning (MAML) [Finn et al., 2017] to various deep anomaly detection models, including one-class classification [Frikha et al., 2021], generative adversarial networks [Lu et al., 2020a], autoencoder [Wu et al., 2021a], graph deviation networks [Ding et al., 2021], and supervised classifiers [Zhang et al., 2020, Feng et al., 2021b]. Some approaches extend prototypical networks to few-shot anomaly detection [Kruspe, 2019, Chen et al., 2022]. Koz-

erawski and Turk [2018] learn a linear SVM with a few samples on top of a frozen pre-trained feature extractor, while Sheynin et al. [2021] learn a hierarchical generative model from a few normal samples for image anomaly detection. Wang et al. [2022] learn an energy model for anomaly detection. The anomalies are scored by the error of reconstructing their embeddings from a set of normal features that are adapted with a few test samples. Huang et al. [2022] learn a category-agnostic model with multiple training categories (a meta set). At test time, a few normal samples from a novel category are used to establish an anomaly detector in the feature space. Huang et al. [2022] does not exploit the presence of a meta-set to learn a stronger anomaly detector through synthetic outlier exposure. While meta-training for object-level anomaly detection (e.g., [Frikha et al., 2021]) is generally simpler (it is easy to find anomaly examples, i.e., other objects different from the normal one), meta-training for anomaly segmentation (e.g., [Huang et al., 2022]) poses a harder task since image defects may differ from object to object (e.g., defects in transistors may not easily generalize to subtle defects in wood textures). Our experiments found that using images from different distributions as example anomalies during training is helpful for anomaly segmentation on MVTec-AD (see Appendix D.9.5).

In contrast to all of the existing few-shot anomaly detection methods, we propose a zero-shot anomaly detection method and demonstrate that the learned anomaly detection model can adapt itself to new tasks without any support samples.

Zero-shot Anomaly Detection. Foundation models pre-trained on massive training samples have achieved remarkable results on zero-shot tasks on images [Radford et al., 2021, Yu et al., 2022, Jia et al., 2021, Yuan et al., 2021]. For example, contrastive language-image pre-training (CLIP) [Radford et al., 2021] is a pre-trained language-vision model learned by aligning images and their paired text descriptions. One can achieve zero-shot image classification with CLIP by searching for the best-aligned text description of the test im-

ages. Esmaeilpour et al. [2022] extend CLIP with a learnable text description generator for out-of-distribution detection. Liznerski et al. [2022] apply CLIP for zero-shot anomaly detection and score the anomalies by comparing the alignment of test images with the correct text description of normal samples. In terms of anomaly segmentation, Trans-MM [Chefer et al., 2021] is an interpretation method for Transformer-based architectures. Trans-MM uses the attention map to generate pixel-level masks of input images, which can be applied to CLIP. MaskCLIP [Zhou et al., 2021] directly exploits CLIP's Transformer layer potential in semantic segmentation to generate pixel-level predictions given class descriptions. MAEDAY [Schwartz et al., 2022] uses the reconstruction error of a pre-trained masked autoencoder [He et al., 2022] to generate anomaly segmentation masks. WinCLIP [Jeong et al., 2023], again using CLIP, slides a window over an image and inspects each patch to detect local defects defined by text descriptions.

However, foundation models have two constraints that do not exist in ACR. First, foundation models are not available for all data types. Foundation models do not exist for example for tabular data, which occurs widely in practice, for example in applications such as network security and industrial fault detection. Also, existing adaptations of foundation models for AD (e.g., CLIP) may generalize poorly to specific domains that have not been covered in their massive training samples. For example, Liznerski et al. [2022] observed that CLIP performs poorly on non-natural images, such as MNIST digits. In contrast, ACR does not rely on a powerful pre-trained foundation model, enabling zero-shot AD on various data types. Second, human involvement is required for foundation models. While previous pre-trained CLIP-based zero-shot AD methods adapt to new tasks through informative prompts given by human experts, our method enriches the zero-shot AD toolbox with a new adaptation strategy without human involvement. Our approach allows the anomaly detector to infer the new task/distribution based on a mini-batch of samples.

Connections to Other Areas. Our problem setup and assumptions share similarities with other research areas but differences are also pronounced. Those areas include *test-time adaptation* [Schneider et al., 2020, Nado et al., 2020, Lim et al., 2023, Wang et al., 2021, Choi et al., 2022], *unsupervised domain adaptation* [Kouw and Loog, 2019], *zero-shot classification* [Xian et al., 2018], *meta-learning* [Finn et al., 2017], and *contextual anomaly detection* [Gupta et al., 2013]. Appendix D.8 details the connections, similarities, and differences.

5.4 Experiments

We evaluate the proposed method ACR on both image (detection/segmentation) and tabular data, where distribution shifts occur at test time. We compare ACR with established baselines based on deep anomaly detection, zero-shot anomaly detection, and few-shot anomaly detection methods. The experiments show that our method is suitable for different data types, applicable to diverse anomaly detection models, robust to various anomaly ratios, and significantly outperforms existing baselines. We report results on image and tabular data in Section 5.4.1 and Section 5.4.2, and ablation studies in Section 5.4.3. Results on more datasets are in Appendices D.9.3 to D.9.6.

5.4.1 Experiments on Images

Visual anomaly detection consists of two major tasks: (image-level) anomaly detection and (pixel-level) anomaly segmentation. The former aims to accurately detect images of abnormal objects, e.g., detecting non-dog images; the latter focuses on detecting pixel-level local defects in an image, e.g., marking board wormholes. We test our method on both tasks and compare it to existing SOTA methods.

Anomaly Detection

We evaluate ACR on images when applied to two simple backbone models: DeepSVDD [Ruff et al., 2018] and a binary classifier. Our method is trained from scratch. The evaluation demonstrates that ACR achieves superior anomaly detection results on corrupted natural images, medical images, and other non-natural images.

Datasets. We study four image datasets: CIFAR100-C [Hendrycks and Dietterich, 2019], OrganA [Yang et al., 2021a] (and MNIST [LeCun et al., 1998], and Omniglot [Lake et al., 2015] in Appendix D.9.4). We consider CIFAR100-C is the noise-corrupted version of CIFAR100's test data, thus considered as distributionally shifted data. We train using all training images from original CIFAR100 and test all models on CIFAR100-C. OrganA is a medical image dataset with 11 classes (for various body organs). We leave two successive classes out for testing and use the other classes for training. We repeat the evaluation on all combinations of two consecutive classes. Across all experiments, we apply the "one-vs-rest" setting at test time, i.e., one class is treated as normal, and all the other classes are abnormal [Ruff et al., 2021]. We report the results averaged over all combinations.

Baselines. We compare our proposed method with a SOTA stationary deep anomaly detector (anomaly detection with an inductive bias (ADIB) [Deecke et al., 2021]), a pre-trained classifier used for batch-level zero-shot anomaly detection (ResNet152 [He et al., 2016]), a SOTA zero-shot anomaly detection baseline (CLIP-AD [Liznerski et al., 2022]), and a few-shot anomaly detection baseline (one-class model-agnostic meta learning (OC-MAML) [Frikha et al., 2021]). ResNet152-I and ResNet152-II differ in the which statistics they use in batch normalization: ResNet152-I uses the statistics from training and ResNet152-II uses the input batch's statistics. See Appendix D.5 for more details.

Implementation Details. We set $\pi = 0.8$ in Equation (5.6) to apply Meta Outlier Exposure. For each approach, we train a single model and test it on different anomaly ratios. Two

backbone models are implemented: DeepSVDD [Ruff et al., 2018] (ACR-DeepSVDD) and a binary classifier with cross entropy loss (ACR-BCE). More details are given in Appendix D.6.

Results. We report the results in terms of the AUROC averaged over five independent test runs with standard deviation. We apply the model to tasks with different anomaly ratios to study the robustness of ACR to the anomaly ratio at test time. Our method ACR significantly outperforms all baselines on Gaussian noise-corrupted CIFAR100-C and OrganA in Table 5.1. In Tables D.5 and D.6 in Appendix D.9, we systematically evaluate all methods on all 19 corrupted versions of CIFAR100 and on non-nature images (MNIST, Omniglot). The results show that on *non-natural* images (OrganA, MNIST, Omniglot) ACR performs the best among all compared methods, including the large pre-trained CLIP-AD baseline; on corrupted *natural* images (CIFAR-100C), ACR achieves results competitive with CLIP-AD and significantly outperforms other baselines. ACR is also robust on various anomaly ratios: without any (hyper)parameter tuning, the results are consistent and don't vary over 3%. The deep anomaly detection baseline, ADIB, doesn't have adaptation ability and thus fails to perform the testing tasks, leading to random guess results. Pre-trained ResNet152 armed with batch normalization layers can adapt but with limited ability, which is in contrast with our method that directly learns to adapt. Few-shot OC-MAML suffers because it requires a large support set at test time to achieve adaptation effectively. CLIP-AD has a strong performance on corrupted natural images but struggles with non-natural images, presumably because it is trained on massive natural images from the internet.

Anomaly Segmentation

We benchmark our method ACR on the MVTec AD dataset [Bergmann et al., 2019] in a zero-shot setup. Experiments show that ACR achieves new state-of-the-art anomaly segmentation performance.

Table 5.1: AUC (%) with standard deviation for anomaly detection on CIFAR100-C with Gaussian noise [Hendrycks and Dietterich, 2019] and medical image dataset, OrganA. ACR with both backbone models perform best.

	CIFAR100-C				OrganA		
	1%	5%	10%	20%	1%	5%	10%
ADIB [Deecke et al., 2021]	50.9±2.4	50.5±0.9	50.6±0.9	50.2±0.5	49.9±6.3	50.3±2.4	50.2±1.3
ResNet152-I [He et al., 2016]	75.6±2.3	73.2±1.3	73.2±0.8	69.9±0.6	54.2±1.1	53.9±0.5	53.2±0.6
ResNet152-II [He et al., 2016]	62.5±3.1	61.8±1.7	61.2±0.6	60.2±0.4	54.2±1.7	53.5±0.8	52.9±0.3
OC-MAML [Frikha et al., 2021]	53.0±3.6	54.1±1.9	55.8±0.6	57.1±1.0	73.7±4.7	72.2±2.6	74.2±2.4
CLIP-AD [Liznerski et al., 2022]	82.3±1.1	82.6±0.9	82.3±0.9	82.6±0.1	52.6±0.8	51.9±0.6	51.5±0.2
ACR-DSVDD (ours)	**87.7±1.4**	**86.3±0.9**	**85.9±0.4**	**85.6±0.4**	79.0±1.0	77.7±0.4	76.3±0.3
ACR-BCE (ours)	84.3±2.2	**86.0±0.3**	**86.0±0.2**	**85.7±0.4**	**81.1±0.8**	**79.5±0.4**	**78.3±0.3**

Table 5.2: Pixel-level and image-level AUC (%) on MVTec AD. On average, our method outperforms the strongest baseline WinCLIP by 7.4% AUC in pixel-level anomaly segmentation.

	MAEDAY [Schwartz et al., 2022]	CLIP [Radford et al., 2021]	Trans-MM [Chefer et al., 2021]	MaskCLIP [Zhou et al., 2021]	WinCLIP [Jeong et al., 2023]	ACR (ours)
pixel-level	69.4	-	57.5±0.0	63.7±0.0	85.1±0.0	**92.5±0.2**
image-level	74.5	74.0±0.0	-	-	**91.8±0.0**	85.8±0.6

Datasets. MVTec AD comprises 15 classes of images for industrial inspection. The goal is to detect the local defects accurately. To implement our method for zero-shot anomaly segmentation tasks, we train on the training sets of all classes except the target one and test on the test set of the target class. For example, when segmenting wormholes on wood boards, we train a model on the other 14 classes' training data except for `wood` and later test on `wood` test set. This satisfies the zero-shot definition as the model doesn't see any `wood` data during training. We apply this procedure for all classes.

Baselines. We compare our method to four zero-shot anomaly segmentation baselines: Trans-MM [Chefer et al., 2021], MaskCLIP [Zhou et al., 2021], MAEDAY [Schwartz et al., 2022], and WinCLIP [Jeong et al., 2023]. The details are described in Section 5.3. We report their results listed in Schwartz et al. [2022], Jeong et al. [2023].

Implementation Details. We first extract informative texture features using a sliding window, which corresponds to 2D convolutions. The convolution kernel is instantiated with the ones in a pre-trained ResNet. We follow the same data pre-processing steps of Cohen

and Hoshen [2020], Rippel et al. [2021], Defard et al. [2021] to extract the features (the third layer's output in our case) of WideResNet-50-2 pre-trained on ImageNet. Second, we detect anomalies in the extracted features in each window position with our ACR method. Specifically, each window position corresponds to one image patch. We stack into a batch the patches taken from a set of images that all share the same spatial position. For example, we may stack the top-left patch of all testing `wood` images into a batch and use ACR to detect anomalies in that batch. Finally, the window-wise anomaly scores are bilinearly interpolated to the original image size to get the pixel-level anomaly scores. In implementing meta outlier exposure, we tried two sources of outliers: one is noise-corrupted images, and the other is images of other classes. We report results of the former in the main paper and the latter in Appendix D.9.5. More implementation details are given in Appendix D.6.

Results. Similar to common practice, we report both the pixel-level and image-level results in Table 5.2. We use the largest pixel-level anomaly score as the image-level score. All methods are evaluated with the AUROC metric. It shows that 1) our method is competitive to the SOTA method in image-level detection tasks, and 2) it surpasses the best baseline WinCLIP by a large margin (7.4% AUC on average) in anomaly segmentation tasks, achieving a new SOTA performance and testifying the potential of our method. We report class-wise results in Appendix D.9.5.

5.4.2 Experiments on Tabular Data

Tabular data is an important data format in many real-world anomaly detection applications, e.g, network intrusion detection and malware detection. Distribution shifts in such data occur naturally over time (e.g., as new malware emerges) and grow over time. Existing zero-shot anomaly detection approaches [Liznerski et al., 2022, Jeong et al., 2023] are not applicable to tabular data. We evaluate ACR on tabular anomaly detection when applied to DeepSVDD

Table 5.3: AUC (%) with standard deviation for anomaly detection on Anoshift with different anomaly contamination rations (1% - 20%) and on different splitting strategies AVG and FAR [Dragoi et al., 2022]. ACR with either backbone model outperforms all baselines. Especially, under the distribution shift occuring in the FAR split, ACR is the only method that is significantly better than random guessing.

	1%		5%		10%		20%	
	FAR	AVG	FAR	AVG	FAR	AVG	FAR	AVG
OC-SVM [Schölkopf et al., 1999]	49.6±0.2	62.6±0.1	49.6±0.2	62.6±0.1	49.5±0.1	62.7±0.1	49.5±0.1	62.6±0.1
IForest [Liu et al., 2012]	25.8±0.4	54.6±0.2	26.1±0.1	54.7±0.1	26.0±0.1	54.6±0.1	26.0±0.1	54.7±0.1
LOF [Breunig et al., 2000]	37.3±0.5	59.6±0.3	37.0±0.1	59.5±0.1	37.0±0.1	59.5±0.1	37.1±0.1	59.5±0.1
KNN [Ramaswamy et al., 2000]	45.0±0.3	70.8±0.1	45.3±0.2	70.9±0.1	45.1±0.1	70.8±0.1	45.2±0.1	70.8±0.1
DSVDD [Ruff et al., 2018]	34.6±0.3	62.3±0.2	34.7±0.1	62.5±0.1	34.7±0.2	62.5±0.1	34.7±0.1	62.5±0.1
AE [Aggarwal, 2017]	18.6±0.2	25.3±0.1	18.7±0.2	25.5±0.1	18.7±0.1	25.5±0.1	18.7±0.1	25.5±0.1
LUNAR [Goodge et al., 2022]	24.5±0.4	38.3±0.4	24.6±0.1	38.6±0.2	24.7±0.1	38.7±0.1	24.6±0.1	38.6±0.1
ICL [Shenkar and Wolf, 2021]	20.6±0.3	50.5±0.2	20.7±0.2	50.4±0.1	20.7±0.1	50.4±0.1	20.8±0.1	50.4±0.1
NTL [Qiu et al., 2021]	40.7±0.3	57.0±0.1	40.9±0.2	57.1±0.1	41.0±0.1	57.1±0.1	41.0±0.1	57.1±0.1
BERT-AD[Dragoi et al., 2022]	28.6±0.3	64.6±0.2	28.7±0.1	64.6±0.1	28.7±0.1	64.6±0.1	28.7±0.1	64.7±0.1
ACR-DSVDD (ours)	62.0±0.5	**74.0±0.2**	61.3±0.1	**73.3±0.1**	60.4±0.1	72.5±0.1	59.1±0.1	71.2±0.1
ACR-NTL (ours)	**62.5±0.2**	73.4±0.1	**62.2±0.1**	**73.2±0.1**	**62.3±0.1**	**73.1±0.1**	**62.0±0.1**	**72.7±0.1**

and NTL. ACR achieves a new SOTA of zero-shot anomaly detection performance on tabular data with temporal distribution shifts.

Datasets. We evaluate all methods on two real-world tabular anomaly detection datasets Anoshift [Dragoi et al., 2022] and Malware [Huynh et al., 2017] where data shifts over time. Anoshift is a data traffic dataset for network intrusion detection collected over ten years (2006-2015). We follow the preprocessing procedure and train/test split suggested in Dragoi et al. [2022]. We train the model on normal data collected from 2006 to 2010 [5], and test on a mixture of normal and abnormal samples (with anomaly ratios varying from 1% to 20%) collected from 2011 to 2015. We also apply similar protocols on Malware [Huynh et al., 2017], a dataset for detecting malicious computer programs, and provide details in Appendix D.9.6.

Baselines. We compare with state-of-the art deep and shallow detectors for tabular anomaly detection [Dragoi et al., 2022, Alvarez et al., 2022, Han et al., 2022] and study their performance under test distribution shifts. The shallow anomaly detection baselines include OC-SVM [Schölkopf et al., 1999], IForest [Liu et al., 2012], LOF [Breunig et al., 2000], and KNN [Ramaswamy et al., 2000]. The deep anomaly detection baselines include DeepSVDD [Ruff

[5] validate on a mixture of normal and abnormal samples collected from 2006 to 2010

et al., 2018], Autoencoder (AE) [Aggarwal, 2017], LUNAR [Goodge et al., 2022], ICL [Shenkar and Wolf, 2021], NTL [Qiu et al., 2021], and BERT-AD [Dragoi et al., 2022]. We adopt the implementations from PyOD [Han et al., 2022] or their official repositories.

Implementation Details. To formulate meta-training sets, we bin the data against their timestamps (year for Anoshift and month for Malware) so each bin corresponds to one training distribution P_j. The training tasks are mixed with normality ratio $\pi = 0.8$. To create more training tasks, we augment the data using attribute permutations, resulting in additional training distributions. These attribute permutations increase the variability of training tasks and encourage the model to learn permutation-invariant features. At test time, the attributes are not permuted. Details are in Appendix D.6.

Results. In Table 5.3, we report the results on Anoshift split into AVG (data from 2011 to 2015) and FAR (data from 2014 and 2015). The two splits show how the performance degrades from average (AVG) to when strong distribution shifts happen after a long time interval (FAR). The results of Malware with varying ratios are in Table D.9 and Appendix D.9.6. We report average AUC with standard deviation over five independent test runs. The results on Anoshift and Malware show that ACR outperforms all baselines on all distribution-shifted settings. Remarkably, ACR is the only method that clearly outperforms random guessing on shifted datasets (the FAR split in Anoshift and the test split in Malware). All baselines perform worse than random on shifted test sets even though they achieve strong results when there are no distribution shifts (see results in Dragoi et al. [2022], Alvarez et al. [2022], Han et al. [2022]). This worse-than-random phenomenon is also verified in the benchmark paper AnoShift [Dragoi et al., 2022]. The reason is that in cyber-security applications (e.g., Anoshift and Malware), the attacks evolve adversarially. The anomalies (cyber attacks) are intentionally updated to be as similar to the normal data to spoof the firewalls. That's why static anomaly detection methods like KNN flip their predictions during test time and achieve worse than random performance. In terms of robustness, although ACR-DeepSVDD's per-

formance degrades a little (within 3%) when the anomaly ratio increases, ACR-NTL is fairly robust to high anomaly ratios. The degradation is attributed to the fact that the majority of normal samples get blurred as the anomaly ratio increase, leading to noisy batch statistics.

5.4.3 Ablation Studies

We perform several ablation studies in Appendix D.9.1, including 1) demonstrating the benefit of the Meta Outlier Exposure loss, 2) studying the effect of batch normalization, and 3) analyzing the effects of the batch sizes and the number of meta-training classes. To show that Meta Outlier Exposure is a favorable option, we compare it against the one-class classification loss and a fine-tuned version of ResNet152 on domain-specific training data. Table D.1 shows that our approach outperforms the two alternatives on two image datasets. To analyze the effect of batch normalization, we adjust batch normalization usage during training and testing listed in Table D.2. More details and the studies on the batch size, the number of meta-training classes, other normalization techniques (LayerNorm, InstanceNorm, and GroupNorm), effects of batch norm position, and robustness of the mixing hyperparameter π can be found in Appendix D.9.1.

5.5 Conclusion

We studied the problem of adapting a learned anomaly detection method to a new data distribution, where the concept of "normality" changed. Our method is a zero-shot approach and requires no training or fine-tuning to a new data set. We developed a new meta-training approach, where we trained an off-the-shelf deep anomaly detection method on a (meta-) set of interrelated datasets, adopting batch normalization in every layer, and used samples from the meta set as either normal samples and anomalies, depending on the context. We

showed that the approach robustly generalized to new, unseen anomalies.

Our experiments on image and tabular data demonstrated superior zero-shot adaptation performance when no foundation model was available. We stress that this is an important result since many, if not most anomaly detection applications in the real world rely on specialized datasets: medical images, data from industrial assembly lines, malware data, network intrusion data etc. Existing foundation models often do not capture these data, as we showed. Ultimately, our analysis shows that relatively small modifications to model training (meta-learning, batch normalization, and providing artificial anomalies from the meta-set) will enable the deployment of existing models in zero-shot anomaly detection tasks.

Limitations & Societal Impacts Our method depends on the three assumptions listed in Section 5.2. If those assumptions are broken, zero-shot adaptation cannot be assured.

Anomaly detectors are trained to detect atypical/under-represented data in a data set. Therefore, deploying an anomaly detector, e.g., in video surveillance, may ultimately discriminate against under-represented groups. Anomaly detection methods should therefore be critically reviewed when deployed on human data.

Chapter 6

Detecting and Adapting to Irregular Distribution Shifts in Bayesian Online Learning

This chapter is based on a published paper at NeurIPS 2021: *Detecting and Adapting to Irregular Distribution Shifts in Bayesian Online Learning* by Aodong Li, Alex Boyd, Padhraic Smyth, Stephan Mandt [Li et al., 2021a]

6.1 Introduction

Deployed machine learning systems are often faced with the problem of distribution shift, where the new data that the model processes is systematically different from the data the system was trained on [Zech et al., 2018, Ovadia et al., 2019]. Furthermore, a shift can happen anytime after deployment, unbeknownst to the users, with dramatic consequences for systems such as self-driving cars, robots, and financial trading algorithms, among many

other examples.

Updating a deployed model on new, representative data can help mitigate these issues and improve general performance in most cases. This task is commonly referred to as *online* or *incremental learning*. Such online learning algorithms face a tradeoff between remembering and adapting. If they adapt too fast, their performance will suffer since adaptation usually implies that the model loses memory of previously encountered training data (which may still be relevant to future predictions). On the other hand, if a model remembers too much, it typically has problems adapting to new data distributions due to its finite capacity.

The tradeoff between adapting and remembering can be elegantly formalized in a Bayesian online learning framework, where a prior distribution is used to keep track of previously learned parameter estimates and their confidences. For instance, variational continual learning (VCL) [Nguyen et al., 2018a] is a popular framework that uses a model's previous posterior distribution as the prior for new data. However, the assumption of such continual learning setups is usually that the data distribution is stationary and not subject to change, in which case adaptation is not an issue.

This paper proposes a new Bayesian online learning framework suitable for non-stationary data distributions. It is based on two assumptions: (i) distribution shifts occur irregularly and must be inferred from the data, and (ii) the model requires not only a good mechanism to aggregate data but also the ability to partially forget information that has become obsolete. To solve both problems, we still use a Bayesian framework for online learning (i.e., letting a previous posterior distribution inform the next prior); however, before combining the previously learned posterior with new data evidence, we introduce an intermediate step. This step allows the model to either broaden the previous posterior's variance to reduce the model's confidence, thus providing more "room" for new information, or remain in the same state (i.e., retain the unchanged, last posterior as the new prior).

We propose a mechanism for enabling this decision by introducing a discrete "change variable" that indicates the model's best estimate of whether the data in the new batch is compatible with the previous data distribution or not; the outcome then informs the Bayesian prior at the next time step. We further augment the scheme by performing beam search on the change variable. This way, we are integrating change detection and Bayesian online learning into a common framework.

We test our framework on a variety of real-world datasets that show concept drift, including basketball player trajectories, malware characteristics, sensor data, and electricity prices. We also study sequential versions of SVHN and CIFAR-10 with covariate drift, where we simulate the shifts in terms of image rotations. Finally, we study word embedding dynamics in an unsupervised learning approach. Our approach leads to a more compact and interpretable latent structure and significantly improved performance in the supervised experiments. Furthermore, it is highly scalable; we demonstrate it on models with hundreds of thousands of parameters and tens of thousands of feature dimensions.

Our paper is structured as follows: we review related work in Section 6.2, introduce our methods in Section 6.3, report our experiments in Section 6.4, and draw conclusions in Section 6.6.

6.2 Related Work

Our paper connects to Bayesian online learning, change detection, and switching dynamical systems.

Bayesian Online and Continual Learning There is a rich existing literature on Bayesian and continual learning. The main challenge in streaming setups is to reduce the impact of

old data on the model which can be done by exponentially decaying old samples [Honkela and Valpola, 2003, Sato, 2001, Graepel et al., 2010] or re-weighting them [McInerney et al., 2015, Theis and Hoffman, 2015]. An alternative approach is to adapt the model posterior between time steps, such as tempering it at a constant rate to accommodate new information [Kulhavỳ and Zarrop, 1993, Kurle et al., 2020]. In contrast, *continual learning* typically assumes a stationary data distribution and simply uses the old posterior as the new prior. A scalable such scheme based on variational inference was proposed by [Nguyen et al., 2018a] which was extended by several authors [Farquhar and Gal, 2018, Schwarz et al., 2018]. A related concept is elastic weight consolidation [Kirkpatrick et al., 2017], where new model parameters are regularized towards old parameter values.

All of these approaches need to make assumptions on the expected frequency and strengh of change which are hard-coded in the model parameters (e.g., exponential decay rates, re-weighting terms, prior strengths, or temperature choices). Our approach, in contrast, detects change based on a discrete variable and makes no assumption about its frequency. Other approaches assume situations where data arrive in irregular time intervals, but are still concerned with static data distributions [Titsias et al., 2019, Lee et al., 2020, Rao et al., 2019].

Change Point Models There is also a rich history of models for change detection. A popular class of change point models includes "product partition models" [Barry and Hartigan, 1992] which assume independence of the data distribution across segments. In this regime, Fearnhead [2005] proposed change detection in the context of regression and generalized it to online inference [Fearnhead and Liu, 2007]; Adams and MacKay [2007] described a Bayesian *online* change point detection scheme (BOCD) based on conditional conjugacy assumptions for one-dimensional sequences. Other work generalized change detection algorithms to multivariate time series [Xuan and Murphy, 2007, Xie et al., 2012] and non-conjugate Bayesian

inference [Saatçi et al., 2010, Knoblauch and Damoulas, 2018, Turner et al., 2013, Knoblauch et al., 2018].

Our approach relies on jointly inferring changes in the data distribution while carrying out Bayesian parameter updates for adaptation. To this end, we detect change in the high-dimensional space of model (e.g., neural network) parameters, as opposed to directly in the data space. Furthermore, a detected change only *partially* resets the model parameters, as opposed to triggering a complete reset.

Titsias et al. [2020] proposed change detection to detect distribution shifts in sequences based on low-dimensional summary statistics such as a loss function; however, the proposed framework does not use an informative prior but requires complete retraining.

Switching Linear Dynamical Systems Since our approach integrates a discrete change variable, it is also loosely connected to the topic of switching linear dynamical systems. Linderman et al. [2017] considered *recurrent* switching linear dynamical systems, relying on Bayesian conjugacy and closed-form message passing updates. Becker-Ehmck et al. [2019] proposed a variational Bayes filtering framework for switching linear dynamical systems. Murphy [2012] and Barber [2012] developed an inference method using a Gaussian sum filter. Instead, we focus on inferring the full history of discrete latent variable values instead of just the most recent one.

Bracegirdle and Barber [2011] introduce a *reset* variable that sets the continuous latent variable to an unconditional prior. It is similar to our work, but relies on using low-dimensional, tractable models. Our tempering prior can be seen as a partial reset, augmented with beam search. We also extend the scope of switching dynamical systems by integrating them into a supervised learning framework.

6.3 Methods

Overview Section 6.3.1 introduces the setup and the novel model structure under consideration. Section 6.3.2 introduces an exact inference scheme based on beam search. Finally, we introduce the variational inference extension for intractable likelihood models in Section 6.3.3.

6.3.1 Problem Assumptions and Structure

We consider a stream of data that arrives in batches $\mathbf{x}_t$ at discrete times t.[1] For supervised setups, we consider pairs of features and targets $(\mathbf{x}_t, \mathbf{y}_t)$, where the task is to model $p(\mathbf{y}_t|\mathbf{x}_t)$. An example model could be a Bayesian neural network, and the parameters $\mathbf{z}_t$ could be the network weights. For notational simplicity we focus on the unsupervised case, where the task is to model $p(\mathbf{x}_t)$ using a model $p(\mathbf{x}_t|\mathbf{z}_t)$ with parameters $\mathbf{z}_t$ that we would like to tune to each new batch.[2] We then measure the prediction error either on one-step-ahead samples or using a held-out test set.

Furthermore, we assume that while the $\mathbf{x}_t$ are i.i.d. within batches, they are not necessarily i.i.d. across batches as they come from a time-varying distribution $p_t(\mathbf{x}_t)$ (or $p_t(\mathbf{x}_t, \mathbf{y}_t)$ in the supervised cases) which is subject to distribution shifts. We do not assume whether these distribution shifts occur instantaneously or gradually. The challenge is to optimally adapt the parameters $\mathbf{z}_t$ to each new batch while borrowing statistical strength from previous batches.

As follows, we will construct a Bayesian online learning scheme that accounts for changes in the data distribution. For every new batch of data, our scheme tests whether the new batch is compatible with the old data distribution, or more plausible under the assumption of a

[1] In an extreme case, it is possible for a batch to include only a single data point.

[2] In supervised setups, we consider a conditional model $p(\mathbf{y}_t|\mathbf{z}_t, \mathbf{x}_t)$ with features $\mathbf{x}_t$ and targets $\mathbf{y}_t$.

change. To this end, we employ a binary "change variable" s_t, with $s_t = 0$ for no detected change and $s_t = 1$ for a detected change. Our model's joint distribution factorizes as follows:

$$p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}, s_{1:T}) = \prod_{t=1}^{T} p(\mathbf{x}_t|\mathbf{z}_t)p(\mathbf{z}_t|s_t; \boldsymbol{\tau}_t)p(s_t). \tag{6.1}$$

We assumed a factorized Bernoulli prior $\prod_t p(s_t)$ over the change variable: an assumption that will simplify the inference, but which can be relaxed. As a result, our model is fully-factorized over time, however, the model can still capture temporal dependencies through the informative prior $p(\mathbf{z}_t|s_t; \boldsymbol{\tau}_t)$. Temporal information enters this prior through certain *sufficient statistics* $\boldsymbol{\tau}_t$ that depend on properties of the previous time-step's approximate posterior.

In more detail, $\boldsymbol{\tau}_t$ is a *functional* on the previous time step's approximate posterior, $\boldsymbol{\tau}_t = \mathcal{F}[p(\mathbf{z}_{t-1}|\mathbf{x}_{1:t-1}, s_{1:t-1})]$.[3] Throughout this paper, we will use a specific form of $\boldsymbol{\tau}_t$, namely capturing the previous posterior's mean and variance.[4] More formally,

$$\boldsymbol{\tau}_t \equiv \{\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}\} \equiv \{\text{Mean}, \text{Var}\}[\mathbf{z}_{t-1}|\mathbf{x}_{1:t-1}, s_{1:t-1}]. \tag{6.2}$$

Based on this choice, we define the conditional prior as follows:

$$p(\mathbf{z}_t|s_t; \boldsymbol{\tau}_t) = \begin{cases} \mathcal{N}(\mathbf{z}_t; \boldsymbol{\mu}_{t-1}, \Sigma_{t-1}) & \text{for } s_t = 0 \\ \mathcal{N}(\mathbf{z}_t; \boldsymbol{\mu}_{t-1}, \beta^{-1}\Sigma_{t-1}) & \text{for } s_t = 1 \end{cases} \tag{6.3}$$

Above, $0 < \beta < 1$ is a hyperparameter referred to as *inverse temperature*[5]. If no change is detected (i.e., $s_t = 0$), our prior becomes a Gaussian distribution centered around the

[3] Subscripts $1:t-1$ indicates the integers from 1 to $t-1$ inclusively.

[4] In later sections, we will use a Gaussian approximation to the posterior, but here it is enough to assume that these quantities are computable.

[5] In general it only requires $\beta > 0$ to be inverse temperature. We further assume $\beta < 1$ in this paper as this value interval broadens and weakens the previous posterior. See the following paragraphs.

previous posterior's mean and variance. In particular, if the previous posterior was already Gaussian, it becomes the new prior. In contrast, if a change was detected, the *broadened* posterior becomes the new prior.

For as long as no changes are detected ($s_t = 0$), the procedure results in a simple Bayesian online learning procedure, where the posterior uncertainty shrinks with every new observation. In contrast, if a change is detected ($s_t = 1$), an overconfident prior would be harmful for learning as the model needs to adapt to the new data distribution. We therefore weaken the prior through *tempering*. Given a temperature β, we raise the previous posterior's Gaussian approximation to the power β, renormalize it, and then use it as a prior for the current time step.

The process of tempering the Gaussian posterior approximation can be understood as removing equal amounts of information in any direction in the latent space. To see this, let $\mathbf{z}$ be a multivariate Gaussian with covariance Σ and $\mathbf{u}$ be a unit direction vector. Then tempering removes an equal amount of information regardless of $\mathbf{u}$, $H_{\mathbf{u}} = \frac{1}{2}\log(2\pi e\mathbf{u}^\top\Sigma\mathbf{u}) - \frac{1}{2}\log\beta$, erasing learned information to free-up model capacity to adjust to the new data distribution. See Supplement E.2 for more details.

Connection to Sequence Modeling Our model assumptions have a resemblance to time series modeling: if we replaced $\boldsymbol{\tau}_t$ with $\mathbf{z}_{t-1}$, we would condition on previous latent states rather than posterior summary statistics. In contrast, our model still factorizes over time and therefore makes weaker assumptions on temporal continuity. Rather than imposing temporal continuity on a data *instance* level, we instead assume temporal continuity at a *distribution* level.

Connection to Changepoint Modeling. We also share similar assumptions with the changepoint literature [Barry and Hartigan, 1992]. However, in most cases, these models

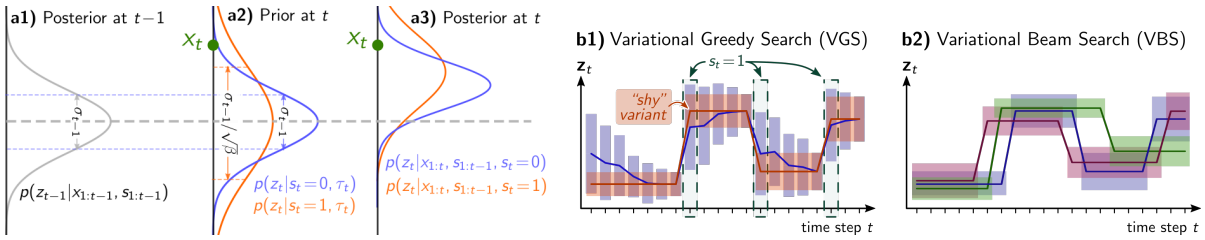


Figure 6.1: **a)** A single inference step for the latent mean in a 1D linear Gaussian model. Starting from the previous posterior (**a1**), we consider both its broadened and un-broadened version (**a2**). Then the model absorbs the observation and updates the priors (**a3**). **b)** Sparse inference via greedy search (**b1**) and variational beam search (**b2**). b) Solid lines indicate fitted mean μ_t over time steps t with boxes representing $\pm 1\sigma$ error bars. See more information about the pictured "shy" variant in Supplement E.3.

don't assume an informative prior, effectively not taking into account any sufficient statistics $\boldsymbol{\tau}_t$. This forces these models to re-learn model parameters from scratch after a detected change, whereas our approach allows for some transfer of information before and after the distribution shift.

6.3.2 Exact Inference

Before presenting a scalable variational inference scheme in our model, we describe an exact inference scheme when everything is tractable, i.e., the special case of linear Gaussian models.

According to our assumptions, the distribution shifts occur at discrete times and are unobserved. Therefore, we have to infer them from the observed data and adapt the model accordingly. Recall the distribution shift is represented by the binary latent variable s_t at time step t. Inferring the posterior over s_t at t will thus notify us how likely the change happens under the model assumption. As follows, we show the posterior of s_t is simple in a tractable model and bears similarity with a likelihood ratio test. Suppose we moved from time step $t-1$ to step t and observed new data $\mathbf{x}_t$. Denote the history decisions and observations by $\{s_{1:t-1}, \mathbf{x}_{1:t-1}\}$, which enters through $\boldsymbol{\tau}_t$. Then by Bayes rule, the exact posterior

over s_t is again a Bernoulli, $p(s_t|s_{1:t-1}, \mathbf{x}_{1:t}) = \text{Bern}(s_t; m)$, with parameter

$$m = \sigma\left(\log \frac{p(\mathbf{x}_t|s_t=1, s_{1:t-1}, \mathbf{x}_{1:t-1})p(s_t=1)}{p(\mathbf{x}_t|s_t=0, s_{1:t-1}, \mathbf{x}_{1:t-1})p(s_t=0)}\right) = \sigma\left(\log \frac{p(\mathbf{x}_t|s_t=1, s_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{x}_t|s_t=0, s_{1:t-1}, \mathbf{x}_{1:t-1})} + \xi_0\right). \tag{6.4}$$

Above, σ is the sigmoid function, and $\xi_0 = \log p(s_t=1) - \log p(s_t=0)$ are the log-odds of the prior $p(s_t)$ and serves as a bias term. $p(\mathbf{x}_t|s_{1:t}, \mathbf{x}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{z}_t)p(\mathbf{z}_t|s_t; \boldsymbol{\tau}_t)d\mathbf{z}_t$ is the model evidence. Overall, m specifies the probability of $s_t = 1$ given $\mathbf{x}_{1:t}$ and $s_{1:t-1}$.

Eq. 6.4 has a simple interpretation as a likelihood ratio test: a change is more or less likely depending on whether or not the observations $\mathbf{x}_t$ are better explained under the assumption of a detected change.

We have described the detection procedure thus far, now we turn to the adaptation procedure. To adjust the model parameters $\mathbf{z}_t$ to the new data given a change or not, we combine the likelihood of $\mathbf{x}_t$ with the conditional prior (Eq. 6.3). This corresponds to the posterior of $\mathbf{z}_t$, $p(\mathbf{z}_t|\mathbf{x}_{1:t}, s_{1:t}) = \frac{p(\mathbf{x}_t|\mathbf{z}_t)p(\mathbf{z}_t|s_t;\boldsymbol{\tau}_t)}{p(\mathbf{x}_t|s_{1:t},\mathbf{x}_{1:t-1})}$, obtained by Bayes rule. The adaptation procedure is illustrated in Fig. 6.1 (a), where we show how a new observation modifies the conditional prior of model parameters.

As a result of Eq. 6.4, we obtain the marginal distribution of $\mathbf{z}_t$ at time t as a binary mixture with mixture weights $p(s_t = 1|s_{1:t-1}, \mathbf{x}_{1:t}) = m$ and $p(s_t = 0|s_{1:t-1}, \mathbf{x}_{1:t}) = 1 - m$: $p(\mathbf{z}_t|s_{1:t-1}, \mathbf{x}_{1:t}) = mp(\mathbf{z}_t|s_t=1, s_{1:t-1}, \mathbf{x}_{1:t}) + (1-m)p(\mathbf{z}_t|s_t=0, s_{1:t-1}, \mathbf{x}_{1:t})$.

Exponential Branching We note that while we had originally started with a posterior $p(\mathbf{z}_{t-1}|\mathbf{x}_{1:t-1}, s_{1:t-1})$ at the previous time, our inference scheme resulted in $p(\mathbf{z}_t|s_{1:t-1}, \mathbf{x}_{1:t})$ being a mixture of two components as it branches over two possible states.[6] When we iterate, we encounter an exponential branching of possibilities, or *hypotheses* over possible sequences

[6]See also Fig. E.1 in Supplement E.3.

of regime shifts $s_{1:t}$. To still carry out the filtering scheme efficiently, we need a truncation scheme, e.g., approximate the bimodal marginal distribution by a unimodal one. As follows, we will discuss two methods—greedy search and beam search—to achieve this goal.

Greedy Search In the simplest "greedy" setup, we train the model in an online fashion by iterating over time steps t. For each t, we update a *truncated* distribution via the following three steps:

1. Compute the conditional prior $p(\mathbf{z}_t|s_t;\boldsymbol{\tau}_t)$ (Eq. 6.3) based on $p(\mathbf{z}_{t-1}|\mathbf{x}_{1:t-1}, s_{1:t-1})$ and evaluate the likelihood $p(\mathbf{x}_t|\mathbf{z}_t)$ upon observing data $\mathbf{x}_t$.

2. Infer whether a change happens or not using the posterior over s_t (Eq. 6.4) and adapt the model parameters $\mathbf{z}_t$ for each case.

3. Select $s_t \in \{0, 1\}$ that has larger posterior probability $p(s_t|s_{1:t-1}, \mathbf{x}_{1:t})$ and its corresponding model hypothesis $p(\mathbf{z}_t|s_{1:t}, \mathbf{x}_{1:t})$ (i.e., make a "hard" decision over s_t with a threshold of $\frac{1}{2}$).

The above filtering algorithm iteratively updates the posterior distribution over $\mathbf{z}_t$ each time it observes new data $\mathbf{x}_t$. In the version of greedy search discussed above, the approach decides immediately, i.e., before observing subsequent data points, whether a change in $\mathbf{z}_t$ has occurred or not in step 3. (Please note the decision is irrelevant to history, as opposed to the beam search described below.) We illustrate greedy search is illustrated in Fig. 6.1 (b1) where VGS is the variational inference counterpart.

Beam Search A greedy search is prone to missing change points in data sets with a low signal/noise ratio per time step because it cannot accumulate evidence for a change point over a series of time steps. The most obvious improvement over greedy search that has the

ability to accumulate evidence for a change point is beam search. Rather than deciding greedily whether a change occurred or not at each time step, beam search considers both cases in parallel as it delays the decision of which one is more likely (see Fig. 6.1 (b2) and Fig. 6.2 (left) for illustration). The algorithm keeps track of a fixed number $K > 1$ of possible hypotheses of change points. For each hypothesis, it iteratively updates the posterior distribution as a greedy search. At time step t, every potential continuation of the K sequences is considered with $s_t \in \{0, 1\}$, thus doubling the number of histories of which the algorithm has to track. To keep the computational requirements bounded, beam search thus discards half of the sequences based on an exploration-exploitation trade-off.

Beam search simultaneously tracks multiple hypotheses necessitating the differentiation between them. In greedy search, we can distinguish hypotheses based on the most recent s_t's value since only two hypotheses are considered at each step. However, beam search considers at most $2K$ hypotheses each step, which exceeds the capacity of a single s_t. We thus resort to the decision history $s_{1:t-1}$ to further tell hypotheses apart. The weight $p(s_{1:t}|\mathbf{x}_{1:t})$ of each hypothesis can be computed recursively:

$$\begin{aligned} p(s_{1:t}|\mathbf{x}_{1:t}) &\propto p(s_t, \mathbf{x}_t|s_{1:t-1}, \mathbf{x}_{1:t-1})p(s_{1:t-1}|\mathbf{x}_{1:t-1}) \\ &\propto p(s_t|s_{1:t-1}, \mathbf{x}_{1:t})p(s_{1:t-1}|\mathbf{x}_{1:t-1}) \end{aligned} \tag{6.5}$$

where the added information $p(s_t|s_{1:t-1}, \mathbf{x}_{1:t})$ at step t is the posterior of s_t (Eq. 6.4). This suggests the "correction in hindsight" nature of beam search: re-ranking the sequence $s_{1:t}$ as a whole at time t indicates the ability to correct decisions before time t.

Another ingredient is a set $\mathbb{B}_t$, which contains the K most probable "histories" $s_{1:t}$ at time t. From time $t-1$ to t, we evaluate the continuation of each hypothesis $s_{1:t-1} \in \mathbb{B}_{t-1}$ as the first two steps of greedy search, leading to $2K$ hypotheses. We then compute the weight of each hypothesis using Eq. 6.5. Finally, select top K hypotheses into $\mathbb{B}_t$ and re-normalize the

weights of hypotheses in $\mathbb{B}_t$.

This concludes the recursion from time $t-1$ to t. With $p(\mathbf{z}_t|s_{1:t}, \mathbf{x}_{1:t})$ and $p(s_{1:t}|\mathbf{x}_{1:t})$, we can achieve any marginal distribution of $\mathbf{z}_t$, such as $p(\mathbf{z}_t|\mathbf{x}_{1:t}) = \sum_{s_{1:t}} p(\mathbf{z}_t|s_{1:t}, \mathbf{x}_{1:t})p(s_{1:t}|\mathbf{x}_{1:t})$.

Beam Search Diversification Empirically, we find that the naive beam search procedure does not realize its full potential. As commonly encountered in beam search, histories over change points are largely shared among all members of the beam. To encourage diverse beams, we constructed the following simple scheme. While transitioning from time $t-1$ to t, every hypothesis splits into two scenarios, one with $s_t = 0$ and one with $s_t = 1$, resulting in $2K$ temporary hypotheses. If two resulting hypotheses only differ in their most recent s_t-value, we say that they come from the same "family." Each member among the $2K$ hypotheses is ranked according to its posterior probability $p(s_{1:t}|\mathbf{x}_{1:t})$ in Eq. 6.5. In a first step, we discard the bottom $1/3$ of the $2K$ hypotheses, leaving $4/3K$ hypotheses (we always take integer multiples of 3 for K). To truncate the beam size from $4/3K$ down to K, we rank the remaining hypotheses according to their posterior probability and pick the top K ones while *also* ensuring that we pick a member from every remaining family. The diversification scheme ensures that underperforming families can survive, leading to a more diverse set of hypotheses. We found this beam diversification scheme to work robustly across a variety of experiments.

6.3.3 Variational Inference

In most practical applications, the evidence term is not available in closed-form, leaving Eq. 6.4 intractable to evaluate. However, we can follow a structured variational inference approach [Wainwright and Jordan, 2008, Hoffman and Blei, 2015, Zhang et al., 2018], defining a joint variational distribution $q(\mathbf{z}_t, s_t|s_{1:t-1}) = q(s_t|s_{1:t-1})q(\mathbf{z}_t|s_{1:t})$, to approximate

$p(\mathbf{z}_t, s_t|s_{1:t-1}, \mathbf{x}_{1:t}) = p(s_t|s_{1:t-1}, \mathbf{x}_{1:t})p(\mathbf{z}_t|s_{1:t}, \mathbf{x}_{1:t})$. This procedure completes the detection and adaptation altogether.

One may wonder how the exact inference schemes for s_t and $\mathbf{z}_t$ are modified in the structured variational inference scenario. In Supplement E.1, we derive the solution for $q(\mathbf{z}_t, s_t|s_{1:t-1})$. Surprisingly we have the following closed-form update equation for $q(s_t|s_{1:t-1})$ that bears strong similarities to Eq. 6.4. The new scheme simply replaces the intractable evidence term with a lower bound proxy – optimized conditional evidence lower bound $\mathcal{L}(q^*|s_{1:t})$ (CELBO, defined later), giving the update

$$q^*(s_t|s_{1:t-1}) = \text{Bern}(s_t; m); \quad m = \sigma\left(\tfrac{1}{T}\mathcal{L}(q^*|s_t{=}1, s_{1:t-1}) - \tfrac{1}{T}\mathcal{L}(q^*|s_t{=}0, s_{1:t-1}) + \xi_0\right). \quad (6.6)$$

Above, we introduced a parameter $T \geq 1$ (not to be confused with β) to optionally downweigh the data evidence relative to the prior (see Experiments Section 3.4).

Now we define the CELBO. To approximate $p(\mathbf{z}_t|s_{1:t}, \mathbf{x}_{1:t})$ by variational distribution $q(\mathbf{z}_t|s_{1:t})$, we minimize the KL divergence between $q(\mathbf{z}_t|s_{1:t})$ and $p(\mathbf{z}_t|s_{1:t}, \mathbf{x}_{1:t})$, leading to

$$q^*(\mathbf{z}_t|s_{1:t}) = \underset{q(\mathbf{z}_t|s_{1:t})\in Q}{\arg\max}\ \mathcal{L}(q|s_{1:t}), \quad (6.7)$$

$$\mathcal{L}(q|s_{1:t}) := \mathbb{E}_q[\log p(\mathbf{x}_t|\mathbf{z}_t)] - \text{KL}(q(\mathbf{z}_t|s_{1:t})||p(\mathbf{z}_t|s_t; \boldsymbol{\tau}_t)).$$

Q denotes the variational family (i.e., factorized normal distributions), and we term $\mathcal{L}(q|s_{1:t})$ CELBO.

The greedy search and beam search schemes also apply to variational inference. We name them *variational greedy search* (VGS, VBS (K=1)) and *variational beam search* (VBS) (Fig. 6.1 (b)).

Algorithm Complexity VBS's computational time and space complexity scale *linearly* with the beam size K. As such, its computational cost is only about $2K$ times larger than greedy search[7]. Furthermore, our algorithm's complexity is $O(1)$ in the sequence length t. It is not necessary to store sequences $s_{1:t}$ as they are just symbols to distinguish hypotheses. The only exception to this scaling would be an application asking for the most likely changepoint sequence in hindsight. In this case, the changepoint sequence (but not the associated model parameters) would need to be stored, incurring a cost of storing exactly $K \times T$ binary variables. This storage is, however, not necessary when the focus is only on adapting to distribution shifts.

6.4 Experiments

Overview The objective of our experiments is to show that, compared to other methods, variational beam search (1) better reacts to different distribution shifts, e.g., *concept drifts* and *covariate drifts*, while (2) revealing interpretable and temporally sparse latent structure. We experiment on artificial data to demonstrate the "correct in hindsight" nature of VBS (Section 6.4.1), evaluate online linear regression on three datasets with concept shifts (Section 6.4.3), visualize the detected change points on basketball player movements, demonstrate the robustness of the hyperparameter β (Section 6.4.3), study Bayesian deep learning approaches on sequences of transformed images with covariate shifts (Section 6.4.4), and study the dynamics of word embeddings on historical text corpora (Section 6.4.5). Unstated experimental details are in Supplement E.7.

[7]This also applies to the baselines "Bayesian Forgetting" (BF) and Variational Continual Learning" (VCL) introduced in Section 6.4.2.

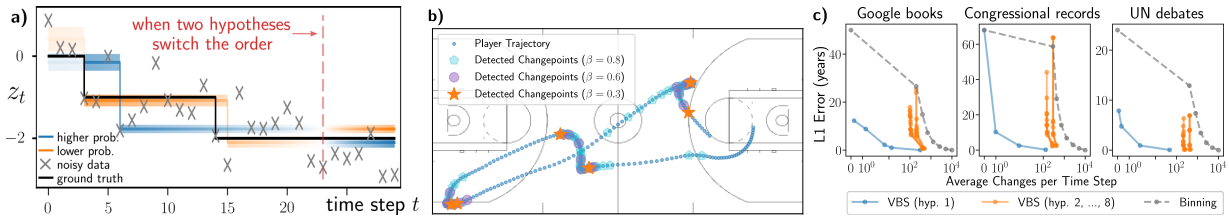


Figure 6.2: **a)** Inferring the mean (black line) of a time-varying data distribution (black samples) with VBS. The initially unlikely hypothesis begins dominating over the other at step 23. **b)** Basketball player tracking: ablation study over β for VBS while fixing other parameters. We used greedy search (K=1) and run the model under different β values. Increasing β leads to more sensitivity to changes in data, leading to more detected changepoints. **c)** Document dating error as a function of model sparsity, measured in average words update per year. As semantic changes get successively sparsified by varying ξ_0 (Eq. 6.6), VBS maintains a better document dating performance compared to baselines.

6.4.1 An Illustrative Example

We first aim to dynamically demonstrate the "correction in hindsight" nature of VBS based on a simple setup involving artificial data. To this end, we formulated a problem of tracking the shifting mean of data samples. This shifting mean is a piecewise-constant step function involving two steps (seen as in black in Fig. 6.2 (a)), and we simulated noisy data points centered around the mean. We then used VBS with beam size $K = 2$ to fit the same latent mean model that generated the data. The color indicates the ranking among both hypotheses at each moment in time (blue as "more likely" vs. orange as "less likely"). While hypothesis 1 assumes a single distribution shift (initially blue), hypothesis 2 (initially orange) assumes two shifts. We see that hypothesis 1 is initially more likely, but gets over-ruled by the better hypothesis 2 later (note the color swap at step 23).

6.4.2 Baselines

In our supervised experiments (Section 6.4.3 and Section 6.4.4), we compared VBS against adaptive methods, Bayesian online learning baselines, and independent batch learning base-

lines.[8] Among the adaptive methods, we formulated a supervised learning version of Bayesian online changepoint detection (BOCD) [Adams and MacKay, 2007].[9] We also implemented Bayesian forgetting (BF) [Kurle et al., 2020] with convolutional neural networks for proper comparisons. Bayesian online learning baselines include variational continual learning (VCL) [Nguyen et al., 2018a] and Laplace propagation (LP) [Smola et al., 2003, Nguyen et al., 2018a]. Finally, we also adopt a trivial baseline of learning independent regressors/classifiers on each batch in both a Bayesian and non-Bayesian fashion. For VBS and BOCD we always report the most dominant hypothesis. In unsupervised learning experiments, we compared against the online version of word2vec [Mikolov et al., 2013] with a diffusion prior, dynamic word embeddings [Bamler and Mandt, 2017].

6.4.3 Bayesian Linear Regression Experiments

As a simple first version of VBS, we tested an online linear regression setup for which the posterior can be computed analytically. The analytical solution removes the approximation error of the variational inference procedure as well as optimization-related artifacts since closed-form updates are available. Detailed derivations are in Supplement E.4.

Real Datasets with Concept Shifts. We investigated three real-world datasets with *concept shifts*:

- **Malware** This dataset is a collection of 100K malignous and benign computer programs, collected over 44 months [Huynh et al., 2017]. Each program has 482 counting features and a real-valued probability $p \in [0, 1]$ of being malware. We linearly predicted the log-odds.

[8] As a reminder, a "batch" at discrete time t is the dataset available for learning; on the other hand, a "mini-batch" is a small set of data used for computing gradients for stochastic gradient-based optimization.

[9] 1) Although BOCD is mostly applied for unsupervised learning, its application in supervised learning and its underlying model's adaptation to change points are seldom investigated. 2) When the model is non-conjugate, such as Bayesian neural networks, we approximate the log evidence $\log p(y|x)$ by the evidence lower bound.

Table 6.1: Evaluation of Different Datasets

Models	CIFAR-10 (Accuracy)↑	SVHN (Accuracy)↑	Malware (MCAE 10^{-2})↓	SensorDrift (MCAE 10^{-2})↓	Elec2 (MCAE 10^{-2})↓	NBAPlayer (LogLike 10^{-2})↑
VBS (K=6)*	**69.2±0.9**	**89.6±0.5**	**11.61**	**10.53**	7.28	**29.49±3.12**
VBS (K=3)*	68.9±0.9	89.1±0.5	11.65	10.71	7.28	**29.22±2.63**
VBS (K=1)*	68.2±0.8	88.9±0.5	11.65	10.86	**7.27**	**29.25±2.59**
BOCD (K=6)♯	65.6±0.8	88.2±0.5	12.93	24.34	12.49	22.96±7.42
BOCD (K=3)♯	67.3±0.8	88.8±0.5	12.74	24.31	12.49	20.93±7.83
BF¶	**69.8±0.8**	**89.9±0.5**	11.71	11.40	13.37	24.17±2.29
VCL†	66.7±0.8	88.7±0.5	13.27	24.90	16.59	3.48±25.53
LP‡	62.6±1.0	82.8±0.9	13.27	24.90	16.59	3.48±25.53
IBS	63.7±0.5	85.5±0.7	16.6	27.71	12.48	-44.87±16.88
IBS (Bayes)	64.5±0.3	87.8±0.1	16.6	27.71	12.48	-44.87±16.88

* proposed, ♯ [Adams and MacKay, 2007], ¶ [Kurle et al., 2020]
† [Nguyen et al., 2018], ‡ [Smola et al., 2003], S Independent Batch

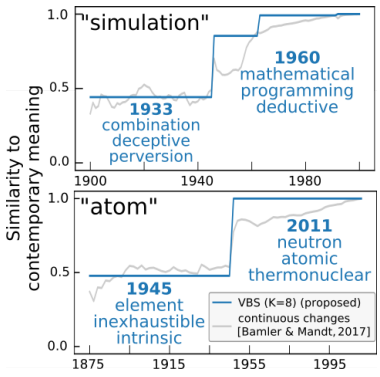


Figure 6.3: Sparse word meaning changes in "simulation" and "atom".

- **SensorDrift** A collection of chemical sensor readings [Vergara et al., 2012]. We predicted the concentration level of gas *acetaldehyde*, whose 2,926 samples and 128 features span 36 months.

- **Elec2** The dataset contains the electricity price over three years of two Australian states [Harries and Wales, 1999]. While the original problem formulation used a majority vote to generate 0-1 binary labels on whether the price increases or not, we averaged the votes out into real-valued probabilities and predicted the log-odds instead. We had 45,263 samples and 14 features.

At each step, only one data sample is revealed to the regressor. We evaluated all methods with one-step-ahead absolute error[10] and computed the mean cumulative absolute error (MCAE) at every step. In Table 6.1, we didn't report the variance of MCAEs since there is no stochastic optimization noise. Table 6.1 shows that VBS has the best average of MCAEs among all methods. We also reported the running performance in Supplement E.7.2, where other experimental details are available as well.

[10]We measured the error in the probability space for classification problems (Malware and Elec2) and the error in the data space for regression problems (SensorDrift).

Basketball Player Tracking. We explored a collection of basketball player movement trajectories.[11] Each trajectory has wide variations in player velocities. We treated the trajectories as time series and used a Bayesian transition matrix to predict the next position $\mathbf{x}_{t+1}$ based on the current position $\mathbf{x}_t$. This matrix is learned and adapted on the fly for each trajectory.

We first investigated the effect of the temperature parameter β in our approach. To this end, we visualized the detected change points on an example trajectory. We used VBS (K=1, greedy search) and compared different values of β in Fig. 6.2 (b). The figure shows that the larger β, the more change points are detected; the smaller β, the detected change points get sparser, i.e., β determines the model's sensitivity to changes. This observation confirms the assumption that β controls the assumed strength of distribution shifts.

In addition, the result also implies the robustness of poorly selected βs. When facing an abrupt change in the trajectory, the regressor has two adapt options based on different βs – make a single strong adaptation or make a sequence of weak adaptations – in either case, the model ends up adapting itself to the new trajectory. In other words, people can choose different β for a specific environment, with a trade-off between adaptation speed and the erased amount of information.

Finally, regarding the quantitative results, we evaluated all methods with the time-averaged predictive log-likelihood on a reserved test set in Table 6.1. Our proposed methods yield better performance than the baselines. In Supplement E.6, we provide more results of change point detection.

[11] `https://github.com/linouk23/NBA-Player-Movements`

6.4.4 Bayesian Deep Learning Experiments

Our larger-scale experiments involve Bayesian convolutional neural networks trained on sequential batches for image classification using CIFAR-10 [Krizhevsky et al., 2009] and SVHN [Netzer et al., 2011]. Every few batches, we manually introduce *covariate shifts* through transforming all images globally by combining rotations, shifts, and scalings. Each transformation is generated from a fixed, predefined distribution (see Supplement E.7.3). The experiment involved 100 batches in sequence, where each batch contained a third of the transformed datasets. We set the temperature $\beta = 2/3$ and set the CELBO temperature $T = 20,000$ (in Eq. 6.6) for all supervised experiments.

Table 6.1 shows the performances of all considered methods and both data sets, averaged across all of the 100 batches. Within their confidence bounds, VBS and BF have comparable performances and outperform the other baselines. We conjecture that the strong performance of BF can be attributed to the fact that our imposed changes are still relatively evenly spaced and regular. The benefit of beam search in VBS is evident, with larger beam sizes consistently performing better.

6.4.5 Unsupervised Experiments

Our final experiment focused on unsupervised learning. We intended to demonstrate that VBS helps uncover interpretable latent structure in high-dimensional time series by detecting change points. We also showed that the detected change points help reduce the storage size and maintain salient features.

Towards this end, we analyzed the semantic changes of individual words over time in an unsupervised setup. We used Dynamic Word Embeddings (DWE) [Bamler and Mandt, 2017] as our base model. The model is an online version of Word2Vec [Mikolov et al., 2013].

Word2Vec projects a vocabulary into an embedding space and measures word similarities by cosine distance in that space. DWE further imposes a time-series prior on the embeddings and tracks them over time. For our proposed approach, we augmented DWE with VBS, allowing us to detect the changes of words meaning.

We analyzed three large time-stamped text corpora, all of which are available online. Our first dataset is the Google Books corpus [Michel et al., 2011] in n-grams form. We focused on 1900 to 2000 with sub-sampled 250M to 300M tokens per year. Second, we used the Congressional Records dataset [Gentzkow et al., 2018], which has 13M to 52M tokens per two-year period from 1875 to 2011. Third, we used the UN General Debates corpus [Jankin Mikhaylov et al., 2017], which has about 250k to 450k tokens per year from 1970 to 2018.

Our first experiments demonstrate VBS provides more interpretable step-wise word meaning shifts than the continuous shifts (DWE). Due to page limits, in Fig. 6.3 we selected two example words and their three nearest neighbors in the embedding space at different years. The evolving nearest neighbors reflect a semantic change of the words. We plotted the most likely hypothesis of VBS in blue and the continuous-path baseline (DWE) in grey. While people can roughly tell the change points from the continuous path, the changes are surrounded by noisy perturbations and sometimes submerged within the noise. VBS, on the other hand, makes clear decisions and outputs explicit change points. As a result, VBS discovers that the word "atom" changes its meaning from "element" to "nuclear" in 1945–the year when two nuclear bombs were detonated; word "simulation" changes its dominant context from "deception" to "programming" with the advent of computers in the 1950s. Besides interpretable changes points, VBS provides multiple plausible hypotheses (Supplement E.7.4).

Our second experiments exemplify the usefulness of the detected *sparse* change points, which lead to sparse segments of embeddings. The usefulness comes in two folds: 1) while alleviating the burden of the disk storage by storing one value for each segment, 2) the sparsity

preserves the salient features of the original model. To illustrate these two aspects, we design a document dating task that exploits the probabilistic interpretation of word embeddings. The idea is to assign a test document to the year whose embeddings provide the highest likelihood. In Figure 6.2 (c), we measure the model sparsity on the x-axis with the average updated embeddings per step (The maximum is 10000, which is the vocabulary size). The feature preservation ability is measured by document dating accuracy on the y-axis. We adjust the prior log-odds ξ_0 (Eq. 6.6) to have successive models with different change point sparsity and then measure the dating accuracy. We also designed an oracle baseline named "binning" (grey, Supplement E.7.4). For VBS, we show the dominant hypothesis (blue) as well as the subleading hypotheses (orange). The most likely hypothesis of VBS outperforms the baseline, leading to higher document dating precision at much smaller disk storage.

6.5 Discussion

Beyond Gaussian Posterior Approximations. While the Gaussian approximation is simple and is widely used (and broadly effective) in practice in Bayesian inference [e.g., Murphy [2012], pp.649-662], our formulation does not rule out the extensions to exponential families. $\boldsymbol{\tau}_t$ in Eq. 6.2 could be generalized by reading off sufficient statistics of the previous approximate posterior. To this end, we need a sufficient statistic that is associated with some measure of entropy or variance that we broaden after each detected change. For example, the Gamma distribution can broaden its scale, and for the categorical distribution, we can increase its entropy/temperature. More intricate (e.g. multimodal) possible alternatives for posterior approximation are also possible, for example, Gaussian mixtures.

6.6 Conclusions

We introduced variational beam search: an approximate inference algorithm for Bayesian online learning on non-stationary data with irregular changes. Our approach mediates the tradeoff between a model's ability to memorize past data while still being able to adapt to change. It is based on a Bayesian treatment of a given model's parameters and aimed at tuning them towards the most recent data batch while exploiting prior knowledge from previous batches. To this end, we introduced a sequence of a discrete change variables whose value controlled the way we regularized the model. For no detected change, we regularized the new learning task towards the previously learned solution; for a detected change, we broadened the prior to give room for new data evidence. This procedure is combined with beam search over the discrete change variables. In different experiments, we showed that our proposed model (1) achieved lower error in supervised setups, and (2) revealed a more interpretable and compressible latent structure in unsupervised experiments.

Broader Impacts. As with many machine learning algorithms, there is a danger that more automation could potentially result in unemployment. Yet, more autonomous adaptation to changes will enhance the safety and robustness of deployed machine learning systems, such as self-driving cars.

Chapter 7

Conclusion

7.1 Technical Summary and Conclusion

Chapters 3 to 5 delve into deep anomaly detection across different scenarios, while Chapter 6 introduces a generic framework for adapting to sequential distribution shifts in supervised and unsupervised learning contexts. We generalize the outlier exposure idea to various settings as shown in the proposed objective functions in Equations (7.1) to (7.3). As follows, we summarize the technical contributions of each chapter.

Chapter 3 Contaminated data–training data containing unnoticed anomalies–is prevalent in the real world. We proposed a new unsupervised approach, Latent Outlier Exposure (LOE), to exploit contaminated data to train a deep anomaly detector. LOE is compatible with various data types and loss functions.

We denoted the loss function used by an anomaly detector by $\mathcal{L}_n^\theta$, intended to be minimized on normal data. For most loss function instances, we can design a complementary loss $\mathcal{L}_a^\theta$ on abnormal data which shares the parameters with $\mathcal{L}_n^\theta$, aimed at

increasing the values of $\mathcal{L}_n^\theta$ on those data. For example, $\mathcal{L}_a^\theta = 1/\mathcal{L}_n^\theta$. By optimizing the two loss functions, we explicitly ensured $\mathcal{L}_n^\theta$ is low on normal data while high on abnormal data. Hendrycks et al. [2018], Ruff et al. [2019] reported the efficacy of the two complementary losses in supervised anomaly detection settings. The supervised setting makes use of the following objective function:

$$\min_\theta \mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(1-y_i)\mathcal{L}_n^\theta(\mathbf{x}_i) + y_i\mathcal{L}_a^\theta(\mathbf{x}_i)$$

where the training dataset comprises labeled data points $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$. $y_i := y(\mathbf{x}_i) \in \{0 := \text{“normal”}, 1 := \text{“abnormal”}\}$ is a binary label indicating whether or not $\mathbf{x}_i$ is normal.

Instead of supervised learning, we considered an unlabeled training dataset $\mathcal{D} := \{\mathbf{x}_i\}_{i=1}^N$ corrupted by unnoticed anomalies with a corruption ratio α (see the generating distribution in Equation (2.1)). The dataset came with no anomaly labels. To represent which datum is potentially anomalous data, we adopted a binary latent variable $\tilde{y}_i \in \{0 := \text{“normal”}, 1 := \text{“abnormal”}\}$. We stressed that the whole model may not be probabilistic. We proposed to solve the constrained optimization problem below to simultaneously learn the model parameters and infer the latent anomaly labels

$$\min_{\tilde{\mathbf{y}}} \min_\theta \mathcal{L}(\theta, \tilde{\mathbf{y}}) = \frac{1}{N}\sum_{i=1}^{N}(1-\tilde{y}_i)\mathcal{L}_n^\theta(\mathbf{x}_i) + \tilde{y}_i\mathcal{L}_a^\theta(\mathbf{x}_i) \tag{7.1}$$
$$\text{such that } \tilde{\mathbf{y}} \in \{0,1\}^N \text{ and } \sum_{i=1}^{N}\tilde{y}_i = \alpha N$$

where α is a hyperparameter, and the constraint guarantees the inferred anomalies constitute a ratio α of the whole dataset.

We further derived a block coordinate descent algorithm to solve the constrained optimization problem Equation (7.1). The algorithm iteratively optimizes the objective function with respect to θ and $\tilde{\mathbf{y}}$ – updating one when fixing the other. We demon-

strated the efficacy of our approach LOE on tabular, image, and video data with multiple backbone models.

Chapter 4 In general, machine learning algorithms, including LOE presented in Chapter 4, cannot correctly classify all data. Further improving the anomaly detection performance needs human feedback. One solution is to actively request some ground-truth anomaly labels from experts. In this direction, we addressed two questions: i) how do we select data points to label? ii) How do we integrate the acquired label information into the model training?

i) How do we select data points to label? We derived a theoretical condition about when the anomaly scores of the labeled data can generalize to unlabeled data. The condition suggests a diverse selection strategy. To this end, we proposed to use the initialization algorithm of k-means++. We made all selections at once, given a budget.

ii) How do we integrate the acquired label information into the model training? Once we collected the labeled data, we proposed to minimize the following semi-supervised outlier exposure loss (SOEL) combined with LOE to incorporate the labeling information,

$$\begin{aligned}\min_{\tilde{\mathbf{y}}} \min_{\theta} \mathcal{L}(\theta, \tilde{\mathbf{y}}) = &\frac{1}{|\mathcal{Q}|} \sum_{j \in \mathcal{Q}} \left(y_j \mathcal{L}_a^\theta(\mathbf{x}_j) + (1 - y_j) \mathcal{L}_n^\theta(\mathbf{x}_j) \right) \\ &+ \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \left(\tilde{y}_i \mathcal{L}_a^\theta(\mathbf{x}_i) + (1 - \tilde{y}_i) \mathcal{L}_n^\theta(\mathbf{x}_i) \right) \\ \text{such that } \tilde{\mathbf{y}} \in \{0,1\}^N &\text{ and } \sum_{i \in \mathcal{U}} \tilde{y}_i + \sum_{j \in \mathcal{Q}} y_j = \alpha N \end{aligned} \tag{7.2}$$

$\mathcal{Q}$ denotes the indices of the selected data for labeling, where the ground-truth binary labels are represented by y_j. $\mathcal{U}$ represents the remaining data indices with no labels. Like LOE, we inferred the data labels in $\mathcal{U}$ during training.

Setting the hyperparameter – anomaly ratio α in Equation (7.1) – requires expert

knowledge. We derived an importance-weighted unbiased estimator of α from the labeled data to eliminate this hyperparameter. This unbiased estimation is nontrivial as the data for labeling are selected in a diversity-driven strategy and are thus not i.i.d. samples that otherwise can be formed for an unbiased estimation.

Chapter 5 We considered zero-shot anomaly detection in a changing environment where the normal data distribution at test time may differ from training. We approached this problem through a simple intuition – consider a thought experiment where a cat photo is mixed with a bunch of photos of sheep. One can quickly identify the anomalous cat image given most sheep images as the context. Motivated by this intuition, we proposed to apply a batch-level prediction strategy and exploited batch normalization layers as a zero-shot adaptation tool to standardize any data batches. Batch normalization layers are prevalent in deep anomaly detection models. Our approach can turn these models into zero-shot adaptive models. To ensure a model uses the batch norm layers to adapt to unseen data distributions at test time, we used meta-training on a meta-dataset during training to promote the zero-shot learning ability.

Suppose we collected a meta-training dataset comprising K different anomaly detection tasks $\{\mathcal{D}_k \sim P_k\}_{k=1}^K$ where P_k denotes the data distribution for task k. We minimized the objective function below

$$\min_{\theta} \mathcal{L}(\theta) = \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{\mathbf{x}_{\mathcal{B}} \sim P_j} \left[\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \left((1 - y_i) \mathbf{L}_{n,i}^{\theta}(\mathbf{x}_{\mathcal{B}}) + y_i \mathbf{L}_{a,i}^{\theta}(\mathbf{x}_{\mathcal{B}}) \right) \right] \tag{7.3}$$

where in practice we sampled iid mini-batches $\mathbf{x}_{\mathcal{B}}$ from the collected dataset $\mathcal{D}_k$ and $\mathcal{B}$ is the index set. $\mathbf{L}_n^{\theta}(\mathbf{x}_{\mathcal{B}})$ (or $\mathbf{L}_a^{\theta}(\mathbf{x}_{\mathcal{B}})$) is a vector of loss functions evaluated on the mini-batch $\mathbf{x}_{\mathcal{B}}$ in a parallelized batch mode with active batch norm layers. Each component of the vectorized loss $\mathbf{L}_{n,i}^{\theta}(\mathbf{x}_{\mathcal{B}})$ (or $\mathbf{L}_{a,i}^{\theta}(\mathbf{x}_{\mathcal{B}})$) corresponds to the loss of an individual data point $\mathbf{x}_{\mathcal{B},i}$, after passing through the deep anomaly detection model along with other data points in the mini-batch. We named the resulting learned representation

automaically centered representation (ACR).

Our method applies to different data types other than images. We reported the first zero-shot anomaly detection result on tabular data and state-of-the-art results on anomaly segmentation.

Chapter 6 We considered updating a model (either in supervised or unsupervised learning[1]) to a sequence of data batches $\{\mathcal{D}_t\}_{t=1}^T$ that arrive at discrete times $t = 1, \ldots, T$. We assumed the data points within each batch are i.i.d.[2] and batches come from piecewise constant data distributions. However, the time points when data distributions shift and the strengths of distribution shifts are unknown.

We proposed a Bayesian framework to simultaneously detect when distribution shifts happen and, as a result, adapt the model. To model the distribution shifts, we introduced a Bernoulli latent "change variable" $s_t \in \{0, 1\}$ at time t indicating whether or not $\mathcal{D}_t$ comes from a different data distribution than the previous $\mathcal{D}_{t-1}$. If there is no shift ($s_t = 0$), we used the posterior distribution at the previous step $t - 1$ as the prior distribution at current step t like Bayesian online learning; if a shift happens ($s_t = 1$), which means the previous posterior distribution is no longer an appropriate prior for the current step, a shift model for parameters θ_t is more desired in this case. We proposed a broadening scheme as the shift model to erase part of the information stored in the previous posterior to allow more room to adapt to the distribution shift.

With the change variable s_t in mind, the idea of detecting the shifts is simple: given the observed data batch $\mathcal{D}_t$, its marginal likelihood $p(\mathcal{D}_t|s_t)$ conditioned on each value of s_t, and the prior distribution $p(s_t)$ at time t, we detected whether a distribution shift happens by inferring the posterior $p(s_t|\mathcal{D}_t)$ through the Bayes rule, which is equivalent to a likelihood-ratio test[3].

[1]The learning settings differ in whether or not data labels are available and which likelihood model is in use.

[2]Each data batch may contain only one data point in the extreme data streaming case.

[3]We omit temporal dependencies in the notation to simplify the illustration.

We derived a scalable variational inference algorithm accompanying our models to jointly infer the posteriors of the model parameters θ_t and the change variables s_t on the fly. Furthermore, the number of model configurations specified by $s_{1:t}$ increases exponentially over time at the speed of $O(2^t)$; we thus proposed to use beam search to trade off the expressivity against computational tractability. We pruned and retained the K most probable model configurations at every time step. Maintaining multiple model configurations with beam search over time allows one to select the correct model in hindsight when more evidence accumulates. We named our method variational beam search (VBS).

In this thesis, we gave an overview of the deep anomaly detection task and its challenges. Our contribution lies in proposing novel learning frameworks for various anomaly detection setups. These setups range from unsupervised training with contaminated data, active and semi-supervised anomaly detection, to zero-shot adaptive anomaly detection. Our frameworks are compatible with varied deep anomaly detection methods and data types. We demonstrated the utilities in images, tabular data, and video experiments. In addition, we proposed a change-point detection and adaptation framework for Bayesian online learning in supervised and unsupervised learning settings.

7.2 Research Outlook

7.2.1 Anomaly Detection with Foundation Models

Foundation models are large deep learning models trained on broad data on the internet. Pre-trained foundation models can be applied to various downstream tasks through few-shot or zero-shot learning. Anomaly detection can be one of the tasks. The literature has seen a surge of interest in language-assisted visual anomaly detection with pre-trained vision-

language model CLIP [Jeong et al., 2023, Zanella et al., 2023, Huang et al., 2024, Zhou et al., 2024]. While CLIP has motivated numerous new anomaly detection applications, its output–numeric anomaly scores–requires threshold and expert explanation. With the development of large language models (LLMs), interactive chatbots have the potential to help explain and characterize anomalies in human languages. Vision anomaly detection can ground the applications on LLMs combined with a vision module such as LLaVA [Liu et al., 2023] so that the model can explain to users in text which elements in an image make the image abnormal. Users can improve the anomaly definition in multi-round interactions to align the detector behavior. The closest work is Gu et al. [2024]. However, Gu et al. [2024] requires normal data during training and does not have zero-shot or few-shot generalization ability.

Tabular data is another important data type in anomaly detection [Rayana, 2016]. Unlike images, tabular data is structured data containing categorical and numerical features designed by experts. Each feature has concrete meanings indicated by the feature names. Large language models (LLMs) rely on understanding the feature names and values to detect factual errors in tabular data [Narayan et al., 2022]. However, in practice, the domains of tabular data can be out of LLM's training data distribution, and the feature values may be pre-processed or anonymous. These two practical situations may break the working assumption of LLMs, causing them to have difficulty understanding the feature names and values, posing challenges to applying LLMs in detecting anomalies for tabular data. A potential solution is to apply retrieval-augmented generation (RAG) [Lewis et al., 2020] to help relate the LLM's generation to a specialized domain knowledge base.

RAG maintains an external knowledge base for a specialized domain. The knowledge base can store related research articles, databases, or user manuals. When a user asks a question, RAG retrieves the question-relevant information from the knowledge base. It then presents the user question and the RAG-retrieved information to the LLM for answering. Using RAG

for tabular anomaly detection requires users to specify the definition of normalcy and requires the RAG-retrieved information to be relevant and applicable to LLMs. One possible retrieved information format is "feature # is important, and it is positively/negatively correlated with the extent of being anomalous."

7.2.2 Continual Anomaly Detection

We presented automatically centered representations (ACR) for zero-shot anomaly detection under distribution shifts in Chapter 5. However, ACR requires batch-level predictions. In a data streaming setup where real-time prediction is valued, assembling a batch may incur a prediction delay. For instance, slow detection of cyber attacks may cause loss of high-stakes information.

A deployed anomaly detection system may face distribution shifts in the data streams. The features related to normality can change over time, and an established detection system may need to be updated. Data shifts can happen in cybersecurity, where cyber attacks evolve adversarially to deceive the detection system. Therefore, a detector should be adaptive to data streams when a distribution shift happens. Even worse, the time when the distribution shift happens can be unobserved. The anomaly detection system should automatically detect changepoints and adapt. This setup is similar to the problem setting that motivates our variational beam search (VBS, in Chapter 6). One potential solution is assuming piece-wise constant data streams and applying VBS in this data streaming setup.

Another complication is that real-world application data streams are mixed with normal data points and anomalies. The occurrence of anomalies may appear to be fake changepoints, which confuses VBS when detecting true distribution shifts. Similarly, the true changepoints may also appear to be anomalies. To address these challenges, we may need to re-design the modeling assumptions of VBS (by augmenting the latent states to incorporate the occurrence

of anomalies) or distinguish the changepoints from anomalies by checking whether or not the changes persist over time. Sankararaman et al. [2022] proposed an anomaly detection method in a similar setup, but their method does not generalize to high-dimensional and complex data.

7.2.3 Anomaly Detection for Scientific Data

We evaluated deep anomaly detection methods on vision and tabular data in the experiments. Some recent research work investigated the application of deep anomaly detection on scientific data types such as water distribution system data and chemical process data [Tian et al., 2023, Hartung et al., 2023]. However, whether or not deep anomaly detection methods generalize to other scientific data types still needs to be studied. One example is detecting heat waves or extreme rainfalls in the climate data. Climate projections rely on physics simulations and demand for massive computations. Due to restricted computation resources, simulation error exists [Yu et al., 2023]. A robust detection method is demanded. One possible approach is to treat the simulated spatial measurements as pixels and measurements along longitudes and latitudes form an "image." Different measurements like temperature or humidity serve as different channels in the "image." Suppose we treat extreme weather events as anomalies in the "image." In that case, our task is to detect those events by segmenting image anomalies. This detection manner is similar to anomaly segmentation in tumor segmentation for medical images and defect segmentation for industrial images. Our methods of Latent Outlier Exposure (LOE) and Automatically Centered Representations (ACR) may play a role. Due to the complexity, impreciseness, and unique textures of the climate data, more domain knowledge for inductive bias and designing appropriate loss functions are anticipated, providing opportunities to develop novel anomaly detection methods.

Bibliography

Ryan Prescott Adams and David JC MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

Charu C Aggarwal. An introduction to outlier analysis. In *Outlier analysis*, pages 1–34. Springer, 2017.

Shikha Agrawal and Jitendra Agrawal. Survey on anomaly detection using data mining techniques. *Procedia Computer Science*, 60:708–713, 2015.

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.

David Arthur and Sergei Vassilvitskii. K-means++ the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035, 2007.

Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*, 2020.

Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. Scalable k-means+. *Proceedings of the VLDB Endowment*, 5(7), 2012.

Robert Bamler and Stephan Mandt. Dynamic word embeddings. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 380–389. JMLR. org, 2017.

David Barber. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

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.

Daniel Barry and John A Hartigan. Product partition models for change point problems. *The Annals of Statistics*, pages 260–279, 1992.

Christoph Baur, Stefan Denner, Benedikt Wiestler, Nassir Navab, and Shadi Albarqouni. Autoencoders for unsupervised anomaly segmentation in brain mr images: a comparative study. *Medical Image Analysis*, 69:101952, 2021.

Jonathan Baxter. A model of inductive bias learning. *Journal of artificial intelligence research*, 12:149–198, 2000.

Philip Becker-Ehmck, Jan Peters, and Patrick Van Der Smagt. Switching linear dynamics for variational bayes filtering. In *International Conference on Machine Learning*, pages 553–562, 2019.

Laura Beggel, Michael Pfeiffer, and Bernd Bischl. Robust anomaly detection in images using adversarial autoencoders. *arXiv preprint arXiv:1901.06355*, 2019.

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

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.

Christopher M Bishop et al. *Neural networks for pattern recognition*. Oxford university press, 1995.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR, 2015.

Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE transactions on pattern analysis and machine intelligence*, 44(11): 7327–7347, 2021.

Chris Bracegirdle and David Barber. Switch-reset models: Exact and approximate inference. In *Proceedings of The Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 190–198, 2011.

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.

Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020. ISSN 2078-2489. doi: 10.3390/info11020125. URL `https://www.mdpi.com/2078-2489/11/2/125`.

Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*, 2019.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

Hila Chefer, Shir Gur, and Lior Wolf. Generic attention-model explainability for interpreting bi-modal and encoder-decoder transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 397–406, 2021.

Bingqing Chen, Luca Bondi, and Samarjit Das. Learning to adapt to domain shifts with few-shot samples in anomalous sound detection. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 133–139. IEEE, 2022.

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

Sungha Choi, Seunghan Yang, Seokeon Choi, and Sungrack Yun. Improving test-time adaptation via shift-agnostic weight regularization and nearest source prototypes. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*, pages 440–458. Springer, 2022.

Gui Citovsky, Giulia DeSalvo, Claudio Gentile, Lazaros Karydas, Anand Rajagopalan, Afshin Rostamizadeh, and Sanjiv Kumar. Batch active learning at scale. *Advances in Neural Information Processing Systems*, 34:11933–11944, 2021.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

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

Corinna Cortes and Mehryar Mohri. Auc optimization vs. error rate minimization. *Advances in neural information processing systems*, 16, 2003.

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.

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.

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.

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.

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.

Allison Del Giorno, J Andrew Bagnell, and Martial Hebert. A discriminative framework for anomaly detection in large videos. In *European Conference on Computer Vision*, pages 334–349. Springer, 2016.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Max Welling Diederik P. Kingma. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

Kaize Ding, Qinghai Zhou, Hanghang Tong, and Huan Liu. Few-shot network anomaly detection via cross-network meta-learning. *Proceedings of the Web Conference 2021*, 2021.

Paul Dokas, Levent Ertoz, Vipin Kumar, Aleksandar Lazarevic, Jaideep Srivastava, and Pang-Ning Tan. Data mining for network intrusion detection. In *Proc. NSF Workshop on Next Generation Data Mining*, pages 21–30. Citeseer, 2002.

Marius Dragoi, Elena Burceanu, Emanuela Haller, Andrei Manolache, and Florin Brad. Anoshift: A distribution shift benchmark for unsupervised anomaly detection. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Sepideh Esmaeilpour, Bing Liu, Eric Robertson, and Lei Shu. Zero-shot out-of-distribution detection based on the pretrained model clip. In *Proceedings of the AAAI conference on artificial intelligence*, 2022.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Generalization of model-agnostic meta-learning algorithms: Recurring and unseen tasks. *Advances in Neural Information Processing Systems*, 34:5469–5480, 2021.

Sebastian Farquhar and Yarin Gal. A unifying bayesian view of continual learning. *Third workshop on Bayesian Deep Learning (NeurIPS 2018)*, 2018.

Paul Fearnhead. Exact bayesian curve fitting and signal segmentation. *IEEE Transactions on Signal Processing*, 53(6):2160–2166, 2005.

Paul Fearnhead and Zhen Liu. Online inference for multiple changepoint problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(4):589–605, 2007.

Jia-Chang Feng, Fa-Ting Hong, and Wei-Shi Zheng. Mist: Multiple instance self-training framework for video anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14009–14018, 2021a.

Tongtong Feng, Q. Qi, Jingyu Wang, and Jianxin Liao. Few-shot class-adaptive anomaly detection with model-agnostic meta-learning. *2021 IFIP Networking Conference (IFIP Networking)*, pages 1–9, 2021b.

Gilberto Fernandes, Joel JPC Rodrigues, Luiz Fernando Carvalho, Jalal F Al-Muhtadi, and Mario Lemes Proença. A comprehensive survey on network anomaly detection. *Telecommunication Systems*, 70:447–489, 2019.

Tharindu Fernando, Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Deep learning for medical anomaly detection–a survey. *ACM Computing Surveys (CSUR)*, 54(7):1–37, 2021.

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.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

Ahmed Frikha, Denis Krompaß, Hans-Georg Köpken, and Volker Tresp. Few-shot one-class classification via meta-learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):7448–7456, May 2021.

Matthew Gentzkow, JM Shapiro, and Matt Taddy. Congressional record for the 43rd-114th congresses: Parsed speeches and phrase counts. In *URL: https://data. stanford. edu/congress text*, 2018.

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.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

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

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Adam Goodge, Bryan Hooi, See-Kiong Ng, and Wee Siong Ng. Lunar: Unifying local outlier detection methods via graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6737–6745, 2022.

Nico Görnitz, Marius Kloft, Konrad Rieck, and Ulf Brefeld. Toward supervised anomaly detection. *Journal of Artificial Intelligence Research*, 46:235–262, 2013.

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.

Thore Graepel, Joaquin Quinonero Candela, Thomas Borchert, and Ralf Herbrich. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *ICML*, 2010.

Zhaopeng Gu, Bingke Zhu, Guibo Zhu, Yingying Chen, Ming Tang, and Jinqiao Wang. Anomalygpt: Detecting industrial anomalies using large vision-language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 1932–1940, 2024.

Manish Gupta, Jing Gao, Charu C Aggarwal, and Jiawei Han. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and data Engineering*, 26(9):2250–2267, 2013.

Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. Adbench: Anomaly detection benchmark. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Michael Harries and New South Wales. Splice-2 comparative evaluation: Electricity pricing. *Citeseer*, 1999.

James Harrison, Apoorva Sharma, Chelsea Finn, and Marco Pavone. Continuous meta-learning without tasks. *Advances in Neural Information Processing Systems*, 33, 2020.

Fabian Hartung, Billy Joe Franks, Tobias Michels, Dennis Wagner, Philipp Liznerski, Steffen Reithermann, Sophie Fellenz, Fabian Jirasek, Maja Rudolph, Daniel Neider, et al. Deep anomaly detection on tennessee eastman process data. *Chemie Ingenieur Technik*, 95(7): 1077–1082, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

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.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

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

Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. *Advances in Neural Information Processing Systems*, 32:15663–15674, 2019.

Waleed Hilal, S Andrew Gadsden, and John Yawney. Financial fraud: a review of anomaly detection techniques and recent advances. *Expert systems With applications*, 193:116429, 2022.

Matthew D Hoffman and David M Blei. Structured stochastic variational inference. In *Artificial Intelligence and Statistics*, 2015.

Steven CH Hoi, Rong Jin, and Michael R Lyu. Large-scale text categorization by batch mode active learning. In *Proceedings of the 15th international conference on World Wide Web*, pages 633–642, 2006.

Antti Honkela and Harri Valpola. On-line variational bayesian learning. In *4th International Symposium on Independent Component Analysis and Blind Signal Separation*, 2003.

Chaoqin Huang, Haoyan Guan, Aofan Jiang, Ya Zhang, Michael Spratling, and Yan-Feng Wang. Registration based few-shot anomaly detection. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIV*, pages 303–319. Springer, 2022.

Chaoqin Huang, Aofan Jiang, Jinghao Feng, Ya Zhang, Xinchao Wang, and Yanfeng Wang. Adapting visual-language models for generalizable anomaly detection in medical images. *CVPR*, 2024.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer, 1992.

Peter J Huber. Robust statistics. In *International encyclopedia of statistical science*, pages 1248–1251. Springer, 2011.

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.

Ngoc Anh Huynh, Wee Keong Ng, and Kanishka Ariyapala. A new adaptive learning algorithm and its application to online malware detection. In *International Conference on Discovery Science*, pages 18–32. Springer, 2017.

Ihab F Ilyas and Xu Chu. *Data cleaning*. Morgan & Claypool, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5070–5079, 2019.

Slava Jankin Mikhaylov, Alexander Baturo, and Niheer Dasandi. United Nations General Debate Corpus. *Harvard Dataverse*, 2017. doi: 10.7910/DVN/0TJX8Y. URL `https://doi.org/10.7910/DVN/0TJX8Y`.

Jongheon Jeong, Yang Zou, Taewan Kim, Dongqing Zhang, Avinash Ravichandran, and Onkar Dabeer. Winclip: Zero-/few-shot anomaly classification and segmentation. In *CVPR*, 2023.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, pages 82(1):35–45, 1960.

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

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Jeremias Knoblauch and Theodoros Damoulas. Spatio-temporal bayesian on-line changepoint detection with model selection. In *International Conference on Machine Learning*, pages 2718–2727. PMLR, 2018.

Jeremias Knoblauch, Jack Jewson, and Theodoros Damoulas. Doubly robust bayesian inference for non-stationary streaming data with β-divergences. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Wouter M Kouw and Marco Loog. A review of domain adaptation without target labels. *IEEE transactions on pattern analysis and machine intelligence*, 43(3):766–785, 2019.

Jedrzej Kozerawski and Matthew A. Turk. Clear: Cumulative learning for one-shot one-class image recognition. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3446–3455, 2018.

Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability*, 3: 71–104, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Citeseer*, 2009.

Anna Kruspe. One-way prototypical networks. *ArXiv*, abs/1906.00820, 2019.

R Kulhavỳ and Martin B Zarrop. On a general concept of forgetting. *International Journal of Control*, 58(4):905–924, 1993.

Richard Kurle, Botond Cseke, Alexej Klushyn, Patrick van der Smagt, and Stephan Günnemann. Continual learning with bayesian neural networks for non-stationary data. In *International Conference on Learning Representations*, 2020.

Michael H Kutner, Christopher J Nachtsheim, John Neter, and William Li. *Applied linear statistical models*. McGraw-hill, 2005.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. In *International Conference on Learning Representations*, 2020.

Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, volume 3, pages 448–455, 2003.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

Aodong Li, Alex Boyd, Padhraic Smyth, and Stephan Mandt. Detecting and adapting to irregular distribution shifts in bayesian online learning. *Advances in neural information processing systems*, 34:6816–6828, 2021a.

Aodong Li, Chen Qiu, Marius Kloft, Padhraic Smyth, Stephan Mandt, and Maja Rudolph. Deep anomaly detection under labeling budget constraints. In *International Conference on Machine Learning*, pages 19882–19910. PMLR, 2023.

Aodong Li, Chen Qiu, Marius Kloft, Padhraic Smyth, Maja Rudolph, and Stephan Mandt. Zero-shot anomaly detection via batch normalization. *Advances in Neural Information Processing Systems*, 36, 2024.

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, 2021b.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018a.

Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, and Xiyang Hu. Copod: copula-based outlier detection. In *2020 IEEE international conference on data mining (ICDM)*, pages 1118–1123. IEEE, 2020.

Zheng Li, Yue Zhao, Xiyang Hu, Nicola Botta, Cezar Ionescu, and George Chen. Ecod: Unsupervised outlier detection using empirical cumulative distribution functions. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

Zhun Li, ByungSoo Ko, and HoJin Choi. Pseudo-labeling using gaussian process for semi-supervised deep learning. In *2018 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 263–269. IEEE, 2018b.

Hyesu Lim, Byeonggeun Kim, Jaegul Choo, and Sungha Choi. Ttn: A domain-shift aware batch normalization in test-time adaptation. In *The Eleventh International Conference on Learning Representations*, 2023.

Scott Linderman, Matthew Johnson, Andrew Miller, Ryan Adams, David Blei, and Liam Paninski. Bayesian learning and inference in recurrent switching linear dynamical systems. In *Artificial Intelligence and Statistics*, pages 914–922, 2017.

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.

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.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

Yusha Liu, Chun-Liang Li, and Barnabás Póczos. Classifier two sample test for video anomaly detections. In *BMVC*, page 71, 2018.

Victor Livernoche, Vineet Jain, Yashar Hezaveh, and Siamak Ravanbakhsh. On diffusion modeling for anomaly detection. In *International Conference on Learning Representations*, 2024.

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.

Philipp Liznerski, Lukas Ruff, Robert A Vandermeulen, Billy Joe Franks, Klaus Robert Muller, and Marius Kloft. Exposing outlier exposure: What can be learned from few, one, and zero outlier images. *Transactions on Machine Learning Research*, 2022.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6467–6476, 2017.

Yiwei Lu, Frank Yu, Mahesh Kumar Krishna Reddy, and Yang Wang. Few-shot scene-adaptive anomaly detection. In *ECCV*, 2020a.

Yiwei Lu, Frank Yu, Mahesh Kumar Krishna Reddy, and Yang Wang. Few-shot scene-adaptive anomaly detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 125–141. Springer, 2020b.

Alireza Makhzani and Brendan J Frey. Winner-take-all autoencoders. *Advances in neural information processing systems*, 28, 2015.

James McInerney, Rajesh Ranganath, and David Blei. The population posterior and bayesian modeling on streams. *Advances in neural information processing systems*, 28:1153–1161, 2015.

Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K Gray, Joseph P Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, et al. Quantitative analysis of culture using millions of digitized books. *science*, 331(6014):176–182, 2011.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

Sohrab Mokhtari, Alireza Abbaspour, Kang K Yen, and Arman Sargolzaei. A machine learning approach for anomaly detection in industrial control systems based on measurement data. *Electronics*, 10(4):407, 2021.

Kevin P Murphy. *Machine learning: A probabilistic perspective*. MIT press, 2012.

Zachary Nado, Shreyas Padhy, D Sculley, Alexander D'Amour, Balaji Lakshminarayanan, and Jasper Snoek. Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv preprint arXiv:2006.10963*, 2020.

E Nalisnick, A Matsukawa, Y Teh, D Gorur, and B Lakshminarayanan. Do deep generative models know what they don't know? In *International Conference on Learning Representations*, 2019.

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.

Avanika Narayan, Ines Chami, Laurel Orr, and Christopher Ré. Can foundation models wrangle your data? *Proceedings of the VLDB Endowment*, 16(4):738–746, 2022.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. *NIPS workshop on deep learning and unsupervised feature learning. Vol. 2011. No. 5*, 2011.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018a.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018b.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Jin Ning, Leiting Chen, Chuan Zhou, and Yang Wen. Deep active autoencoders for outlier detection. *Neural Processing Letters*, pages 1–13, 2022.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua V Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, 2019.

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.

Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. *ACM computing surveys (CSUR)*, 54(2):1–38, 2021a.

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, 2021b.

Dan Pelleg and Andrew Moore. Active learning for anomaly and rare-category detection. *Advances in neural information processing systems*, 17, 2004.

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.

Robert Pinsler, Jonathan Gordon, Eric Nalisnick, and José Miguel Hernández-Lobato. Bayesian batch active learning as sparse subset approximation. *Advances in neural information processing systems*, 32, 2019.

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.

Chen Qiu, Timo Pfrommer, Marius Kloft, Stephan Mandt, and Maja Rudolph. Neural transformation learning for deep anomaly detection beyond images. In *International Conference on Machine Learning*, pages 8703–8714. PMLR, 2021.

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, 2022a.

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, pages 18153–18167. PMLR, 2022b.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10428–10436, 2020.

Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 427–438, 2000.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822, 2014.

Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. In *Advances in Neural Information Processing Systems*, pages 7647–7657, 2019.

Shebuti Rayana. Odds library, 2016. URL `https://odds.cs.stonybrook.edu`.

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.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

Oliver Rippel, Patrick Mertens, and Dorit Merhof. Modeling the distribution of normal data in pre-trained deep features for anomaly detection. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6726–6733. IEEE, 2021.

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. PMLR, 2018.

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.

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*, 109(5):756–795, 2021.

C Rusmassen and C Williams. Gaussian process for machine learning, 2005.

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.

Yunus Saatçi, Ryan D Turner, and Carl Edward Rasmussen. Gaussian process change point models. In *ICML*, 2010.

Abishek Sankararaman, Balakrishnan Narayanaswamy, Vikramank Y Singh, and Zhao Song. Fitness:(fine tune on new and similar samples) to detect anomalies in streams with drift and outliers. In *International Conference on Machine Learning*, pages 19153–19177. PMLR, 2022.

Masa-Aki Sato. Online model selection based on the variational bayes. *Neural computation*, 13(7):1649–1681, 2001.

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.

Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *Advances in Neural Information Processing Systems*, 33:11539–11551, 2020.

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.

Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 12, 1999.

Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

Eli Schwartz, Assaf Arbelle, Leonid Karlinsky, Sivan Harary, Florian Scheidegger, Sivan Doveh, and Raja Giryes. Maeday: Mae for few and zero shot anomaly-detection. *arXiv preprint arXiv:2211.14307*, 2022.

Jonathan Schwarz, Daniel Altman, Andrew Dudzik, Oriol Vinyals, Yee Whye Teh, and Razvan Pascanu. Towards a natural benchmark for continual learning. In *NeurIPS Workshop on Continual Learning*, 2018.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Dudley Shapere. The structure of scientific revolutions. *The Philosophical Review*, 73(3): 383–394, 1964.

Tom Shenkar and Lior Wolf. Anomaly detection for tabular data with internal contrastive learning. In *International Conference on Learning Representations*, 2021.

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`.

Shelly Sheynin, Sagie Benaim, and Lior Wolf. A hierarchical transformation-discriminating generative model for few shot anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8495–8504, October 2021.

Yaniv Shulman. Unsupervised contextual anomaly detection using joint deep variational generative models. *arXiv preprint arXiv:1904.00548*, 2019.

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.

Alexander J Smola, Vishy Vishwanathan, and Eleazar Eskin. Laplace propagation. In *NIPS*, pages 441–448, 2003.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020a.

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*, 2020b.

Jacob Steinhardt, Pang Wei Koh, and Percy Liang. Certified defenses for data poisoning attacks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 3520–3532, 2017.

Mahito Sugiyama and Karsten Borgwardt. Rapid distance-based outlier detection via sampling. *Advances in Neural Information Processing Systems*, 26:467–475, 2013.

Iwan Syarif, Adam Prugel-Bennett, and Gary Wills. Data mining approaches for network intrusion detection: from dimensionality reduction to misuse and anomaly detection. *Journal of Information Technology Review*, 3(2):70–83, 2012.

Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. *Advances in neural information processing systems*, 33:11839–11852, 2020.

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.

David M. J. Tax and Robert P. W. Duin. Support vector data description. *Mach. Learn.*, 54(1):45–66, 2004a.

David MJ Tax and Robert PW Duin. Support vector data description. *Machine learning*, 54:45–66, 2004b.

Lucas Theis and Matt Hoffman. A trust-region method for stochastic variational inference with applications to streaming data. In *International Conference on Machine Learning*, pages 2503–2511. PMLR, 2015.

Zhiwen Tian, Ming Zhuo, Leyuan Liu, Junyi Chen, and Shijie Zhou. Anomaly detection using spatial and temporal information in multivariate time series. *Scientific Reports*, 13 (1):4400, 2023.

Michalis K Titsias, Jonathan Schwarz, Alexander G de G Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with gaussian processes. In *International Conference on Learning Representations*, 2019.

Michalis K Titsias, Jakub Sygnowski, and Yutian Chen. Sequential changepoint detection in neural networks with checkpoints. *arXiv preprint arXiv:2010.03053*, 2020.

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.

Radu Tudor Ionescu, Sorina Smeureanu, Bogdan Alexe, and Marius Popescu. Unmasking the abnormal events in video. In *Proceedings of the IEEE international conference on computer vision*, pages 2895–2903, 2017.

Ryan Turner, Steven Bottone, and Clay Stanek. Online variational approximations to non-exponential family change point models: With application to radar tracking. In *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 1*, pages 306–314, 2013.

Alexander Vergara, Shankar Vembu, Tuba Ayhan, Margaret A Ryan, Margie L Homer, and Ramón Huerta. Chemical gas sensor drift compensation using classifier ensembles. *Sensors and Actuators B: Chemical*, 166:320–329, 2012.

Bjarni J Vilhjálmsson and Magnus Nordborg. The nature of confounding in genome-wide association studies. *Nature Reviews Genetics*, 14(1):1–2, 2013.

Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021.

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.

Ze Wang, Yipin Zhou, Rui Wang, Tsung-Yu Lin, Ashish Shah, and Ser-Nam Lim. Few-shot fast-adaptive anomaly detection. In *Advances in Neural Information Processing Systems*, 2022.

Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. In *International Conference on Learning Representations*, 2018.

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 (ICCV)*, pages 4369–4378, October 2021a.

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, 2021b.

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.

Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2251–2265, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv*, 2017.

Zehao Xiao, Xiantong Zhen, Shengcai Liao, and Cees GM Snoek. Energy-based test sample adaptation for domain generalization. In *The Eleventh International Conference on Learning Representations*, 2023.

Guoyang Xie, Jinbao Wang, Jiaqi Liu, Jiayi Lyu, Yong Liu, Chengjie Wang, Feng Zheng, and Yaochu Jin. Im-iad: Industrial image anomaly detection benchmark in manufacturing. *IEEE Transactions on Cybernetics*, 2024.

Yao Xie, Jiaji Huang, and Rebecca Willett. Change-point detection for high-dimensional time series with missing data. *IEEE Journal of Selected Topics in Signal Processing*, 7(1): 12–27, 2012.

Xiang Xuan and Kevin Murphy. Modeling changing dependency structure in multivariate time series. In *Proceedings of the 24th international conference on Machine learning*, pages 1055–1062, 2007.

Jiancheng Yang, Rui Shi, and Bingbing Ni. Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis. In *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pages 191–195. IEEE, 2021a.

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*, 2021b.

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.

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.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *Transactions on Machine Learning Research*, 2022.

Sungduk Yu, Walter M Hannah, Liran Peng, Mohamed Aziz Bhouri, Ritwik Gupta, Jerry Lin, Björn Lütjens, Justus C Will, Tom Beucler, Bryce E Harrop, et al. Climsim: An open large-scale dataset for training high-resolution physics emulators in hybrid multi-scale climate simulators. *Advances in Neural Information Processing Systems Track on Datasets and Benchmarks*, 2023.

Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.

Luca Zanella, Benedetta Liberatori, Willi Menapace, Fabio Poiesi, Yiming Wang, and Elisa Ricci. Delving into clip latent space for video anomaly recognition. *arXiv preprint arXiv:2310.02835*, 2023.

John R Zech, Marcus A Badgeley, Manway Liu, Anthony B Costa, Joseph J Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLoS medicine*, 15(11): e1002683, 2018.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *Proceedings of machine learning research*, 70:3987, 2017.

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.

Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8): 2008–2026, 2018.

Shen Zhang, Fei Ye, Bingnan Wang, and Thomas G. Habetler. Few-shot bearing anomaly detection via model-agnostic meta-learning. *2020 23rd International Conference on Electrical Machines and Systems (ICEMS)*, pages 1341–1346, 2020.

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.

Chong Zhou, Chen Change Loy, and Bo Dai. Denseclip: Extract free dense labels from clip. *arXiv preprint arXiv:2112.01071*, 2021.

Qihang Zhou, Guansong Pang, Yu Tian, Shibo He, and Jiming Chen. AnomalyCLIP: Object-agnostic prompt learning for zero-shot anomaly detection. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=buC4E91xZE`.

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.

Appendix A

Chapter 1

Anomalies, Outliers, and Novelties. We borrow the definitions of these concepts from Ruff et al. [2021]. While anomalies, outliers, and novelties all occur with low probabilities, anomalies originate from a different data-generating process from the normal process. Outliers come from the same data-generating process as the normal data. Novelties are a new mode of a non-stationary normal data-generating process.

Anomaly Detection in Other Disciplines. Anomaly detection holds significance within the field of statistics [Kutner et al., 2005, Chapter 3]. Outliers can manifest in predictor or response variables, impacting statistical modeling in two primary ways. Firstly, outliers can pose modeling difficulties to statistical analysis, such as distorting a regression model and causing issues like lack of fit. Secondly, the presence of outliers following a pattern may signal a need to re-evaluate modeling assumptions.

Anomaly detection is also interesting within data mining [Syarif et al., 2012, Dokas et al., 2002, Agrawal and Agrawal, 2015]. When presented with a dataset, the objective is to detect outliers within it, a process often referred to as transductive learning. Conversely, in machine

learning, anomaly detection seeks to grasp the concept of normality from a dataset and then predict whether a potential unseen data point aligns with that notion of normalcy, known as inductive learning. While anomaly detection in data mining has a different goal, the learning procedure often results in a detection model that can predict the label of unseen data.

Connection to Out-of-Distribution (OOD) Detection. OOD detection has different motivations from anomaly detection. OOD detection aims to make a classifier safe. Because classifiers tend to make confident predictions even if the input (or the OOD data) differs from the classifier's training data distribution, OOD detection detects OOD data and rings an alarm for the classifier.

The motivation decides that OOD detection and anomaly detection have different training data resources. While OOD detection has a per-class labeled multi-class training dataset, a pre-trained classifier, or both for training, anomaly detection only has a normal dataset, but any potential fine-grained labels within the normal dataset may be missing.

Appendix B

Chapter 3

B.1 Details on Toy Data Experiments

We generate the toy data with a three-component Gaussian mixture. The normal data is generated from $p_n = \mathcal{N}(\mathbf{x}; [1, 1], 0.07I)$, and the anomalies are sampled from $p_a = \mathcal{N}(\mathbf{x}; [-0.25, 2.5], 0.03I) + \mathcal{N}(\mathbf{x}; [-1., 0.5], 0.03I)$. There are 90 normal samples and 10 abnormal samples. All samples are mixed up as the contaminated training set.

To learn a anomaly detector, we used one-class Deep SVDD [Ruff et al., 2018] to train a one-layer radial basis function (RBF) network where the Gaussian function is used as the RBF. The hidden layer contains three neurons whose centers are fixed at the center of each component and whose scales are optimized during training. The output of the RBF net is a linear combination of the outputs of hidden layers. Here we set the model output to be a 1D scalar, as the projected data representation of Deep SVDD.

For Deep SVDD configuration, we randomly initialized the model center (not to be confused with the center of the Gaussian RBF) and made it learnable during training. We also

added the bias term in the last layer. Although setting a learnable center and adding bias terms are not recommended for Deep SVDD [Ruff et al., 2018] due to the all-zero trivial solution, we found these practices make the model flexible and converge well and learn a much better anomaly detector than vice verse, probably because the random initialization and small learning rate serve as regularization and the model converges to a local optimum before collapses to the trivial solution. During training, we used Adam [Kingma and Ba, 2015] stochastic optimizer and set the mini-batch size to be 25. The learning rate is 0.01, and we trained the model for 200 epochs. The decision boundary in Figure 3.1 plots the 90% fraction of the anomaly scores.

B.2 Baseline Details

Across all experiments, we employ two baselines that do not utilize anomalies to help training 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 actions 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 Eq. 3.1 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 current detection model, then removes top α most likely anomalous samples from the mini-batch. The remaining samples performs the model update. We name the second baseline *Refine*, which still follows Alg. 1 but removes $\mathcal{L}_a^\theta$ in Eq. 3.1. 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 Implementation Details

We apply NTL to all datasets including both visual datasets and tabular datasets. Below we provide the implementation details of NTL on each class of datasets.

NTL on image data NTL is built upon the final pooling layer of a pre-trained ResNet152 on CIFAR-10 and F-MNIST (as suggested in Defard et al. [2021]), and upon the third residual block of a pre-trained WideResNet50 on MVTEC (as suggested in Reiss et al. [2021]). On all image datasets, the pre-trained feature extractors are frozen during training. We set the number of transformations as 15 and use three linear layers with intermediate 1D batchnorm layers and ReLU activations for transformations modelling. The hidden sizes of the transformation networks are $[2048, 2048, 2048]$ on CIFAR-10 and F-MNIST, and $[1024, 1024, 1024]$ on MVTEC. The encoder is one linear layer with units of 256 for CIFAR-10 and MVTEC, and is two linear layers of size $[1024, 256]$ with an intermediate ReLU activation for F-MNIST. On CIFAR-10, we set mini-batch size to be 500, learning rate to be 4e-4, 30 training epochs with Adam optimizer. On F-MNIST, we set mini-batch size to be 500, learning rate to be 2e-4, 30 training epochs with Adam optimizer. On MVTEC, we set mini-batch size to be 40, 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 On all tabular data, we set the number of transformations to 9, use two fully-connected network layers for the transformations and four fully-connected network layers for the encoder. The hidden size of layers in the transformation networks and the encoder is two times the data dimension for low dimensional data, and 64 for high dimensional data. The embedding size is two times the data dimension for low dimensional data, and 32 for high dimensional data. The transformations are either parametrized as

the transformation network directly or a residual connection of the transformation network and the original sample. We search the best-performed transformation parameterization and other hyperparameters based on the performance of the model trained on clean 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 Following the suggestions of Pang et al. [2020], we first extract frame features through a ResNet50 pretrained on ImageNet. The features are sent to an NTL with the same backbone model as used on CIFAR-10 (see NTL on image data) except that 9 transformations are used. Both the ResNet50 and NTL are updated from end to end. During training, we use Adam stochastic optimizer with the batch size set to be 192 and learning rate set 1e-4. We update the model for 3 epochs and report the results with three independent runs.

MHRot on image data MHRot [Hendrycks et al., 2019] applies self-supervised learning on hand-crafted image transformations including rotation, horizontal shift, and vertical shift. The learner learns to solve three different tasks: one for predicting rotation ($r \in \mathcal{R} \equiv \{0°, \pm 90°, 180°\}$), one for predicting vertical shift ($s^v \in \mathcal{S}^v \equiv \{0\,\text{px}, \pm 8\,\text{px}\}$), and one for predicting horizontal shift ($s^h \in \mathcal{S}^h \equiv \{0\,\text{px}, \pm 8\,\text{px}\}$). We define the composition of rotation, vertical shift, and horizontal shift as $T \in \mathcal{T} \equiv \{r \circ s^v \circ s^h \mid r \in \mathcal{R}, s^v \in \mathcal{S}^v, s^h \in \mathcal{S}^h\}$. We also define the head labels $t_k^1 = r_a, t_k^2 = s_b^v, t_k^3 = s_c^h$ for a specific composed transformation $T_k = r_a \circ s_b^v \circ s_c^h$. Overall, there are 36 transformations.

We implement the model on the top of GOAD [Bergman and Hoshen, 2020], a similar self-supervised anomaly detector. The backbone model is a WideResNet16-4. Anomaly scores is used for ranking in the mini-batch in pseudo label assignments. For F-MNIST, we use

$\mathcal{L}_n^\theta$, the normality training loss, as the anomaly score. For CIFAR-10, we find that using a separate anomaly score mentioned in [Bergman and Hoshen, 2020] leads to much better results than the original training loss anomaly score.

During training, we set mini-batch size to be 10, learning rate to be 1e-3 for CIFAR-10 and 1e-4 for F-MNIST, 16 training epochs for CIFAR-10 and 3 training epochs for F-MNIST with Adam optimizer. We report the results with 3-5 independent runs.

B.4 Additional Experimental Results

We provide additional results of the experiments on tabular datasets. We report the F1-scores in Table B.1 and the AUCs in Table B.2. The number in the brackets is the average performance difference from the model trained on clean data. Remarkably, on some datasets, LOE trained on contaminated data can achieve better results than on clean data (as shown in Tables B.1 and B.2), suggesting that the latent anomalies provide a positive learning signal. Overall, we can see that LOE improves the performance of anomaly detection methods on contaminated tabular datasets significantly.

Table B.1: F1-score (%) with standard deviation for anomaly detection on 30 tabular datasets which are from the empirical study of Shenkar and Wolf [2022]. For all experiments, we set the contamination ratio of the training set as 10%. The number in the brackets is the average performance difference from the model trained on clean data. LOE outperforms the "Blind" and "Refine" baselines.

	NTL				ICL			
	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)	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)**	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	29.7±3.5 (-21.6)	42.7±7.1 (-8.6)	47.7±11.4 (-3.6)	**50.3±4.5** **(-1.0)**	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	57.6±2.5 (-3.0)	59.1±2.1 (-1.5)	62.1±2.8 (+1.5)	**62.7±3.3** **(+2.1)**	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	84.0±1.8 (-8.4)	93.1±0.9 (+0.7)	**95.6±0.4** **(+3.2)**	95.3±0.4 (+2.9)	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	21.8±4.9 (-35.0)	45.2±7.9 (-11.6)	**73.0±7.9** **(+16.2)**	57.8±5.5 (+1.0)	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	0.0±0.0 (-95.6)	88.9±14.1 (-6.7)	**100±0.0** **(+4.4)**	**100±0.0** **(+4.4)**	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 cover	20.4±4.0 (-44.2)	56.2±4.9 (-8.4)	61.1±34.9 (-3.5)	**67.6±30.6** **(+3.0)**	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	11.1±7.0 (-6.7)	15.6±5.4 (-2.2)	17.8±5.4 (+0.0)	**20.0±8.3** **(+2.2)**	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	89.0±1.5 (-3.5)	91.0±2.0 (-1.5)	91.0±1.7 (-1.5)	**91.3±2.2** **(-1.2)**	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	95.9±0.0 (-2.4)	96.0±1.1 (-2.3)	98.1±0.4 (-0.2)	**98.4±0.1** **(+0.1)**	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	98.4±0.1 (+0.2)	98.4±0.2 (+0.2)	89.1±1.7 (-9.1)	**98.6±0.0** **(+0.4)**	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	36.4±3.6 (-11.0)	44.4±3.1 (-3.0)	25.4±10.0 (-22.0)	**45.6±10.6** **(-1.8)**	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	53.3±12.5 (-20.0)	60.0±8.2 (-13.3)	60.0±13.3 (-13.3)	**73.3±22.6** **(+0.0)**	43.3±8.2 (-40.0)	60.0±8.2 (-23.3)	80.0±12.5 (-3.3)	**83.3±10.5** **(+0.0)**
mammogra.	5.5±2.8 (-21.3)	2.6±1.7 (-24.2)	3.3±1.6 (-23.5)	**13.5±3.8** **(-13.3)**	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 tabular	78.6±0.5 (-6.6)	**80.3±1.1** **(-4.9)**	71.8±1.8 (-13.4)	76.3±2.1 (-8.9)	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	45.5±9.6 (-50.5)	**58.2±3.5** **(-37.8)**	**58.2±6.2** **(-37.8)**	50.1±8.9 (-45.9)	70.4±13.4 (-29.6)	94.4±6.3 (-5.6)	**100±0.0** **(+0.0)**	99.9±0.1 (-0.1)
musk	21.0±3.3 (-79.0)	98.8±0.4 (-1.2)	**100±0.0** **(+0.0)**	**100±0.0** **(+0.0)**	6.2±3.0 (-93.8)	**100±0.0** **(+0.0)**	**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)**	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	5.0±2.5 (-56.3)	32.6±10.0 (-28.7)	79.4±4.7 (+18.1)	**81.9±4.3** **(+20.6)**	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	60.3±2.6 (-1.2)	61.0±1.9 (-0.5)	**61.3±2.4** **(-0.2)**	61.0±0.9 (-0.5)	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	73.6±0.4 (-1.0)	74.1±0.3 (-0.5)	**74.8±0.4** **(+0.2)**	74.7±0.1 (+0.1)	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	26.8±1.5 (-65.2)	86.8±4.0 (-5.2)	90.7±1.1 (-1.3)	**91.0±0.7** **(-1.0)**	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	11.9±1.8 (-0.6)	11.5±1.0 (-1.0)	**18.1±0.7** **(+5.6)**	17.1±0.6 (+4.6)	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	97.0±0.3 (+0.3)	97.0±0.2 (+0.3)	**97.1±0.2** **(+0.4)**	97.0±0.2 (+0.3)	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	6.9±1.2 (-2.6)	8.2±2.1 (-1.3)	43.3±5.6 (+33.8)	**50.8±2.5** **(+41.3)**	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	43.4±5.5 (-34.4)	55.1±4.2 (-22.7)	**82.4±2.7** **(+4.6)**	**82.4±2.3** **(+4.6)**	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	22.0±4.5 (-8.7)	21.3±4.5 (-9.4)	22.7±11.0 (-8.0)	**25.3±4.0** **(-5.4)**	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	36.0±1.8 (-40.7)	50.4±8.8 (-26.3)	**62.8±9.5** **(-13.9)**	48.4±6.6 (-28.3)	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	25.7±12.3 (-39.1)	45.7±15.5 (-19.1)	**76.2±6.0** **(+11.4)**	69.5±3.8 (+4.7)	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	24.0±18.5 (-68.0)	66.0±12.0 (-26.0)	90.0±0.0 (-2.0)	**92.0±4.0** **(+0.0)**	4.0±4.9 (-86.0)	10.0±8.9 (-80.0)	98.0±4.0 (+8.0)	**100±0.0** **(+10.0)**

Table B.2: AUC (%) with standard deviation for anomaly detection on 30 tabular datasets which are from the empirical study of Shenkar and Wolf [2022]. For all experiments, we set the contamination ratio of the training set as 10%. The number in the brackets is the average performance difference from the model trained on clean data. LOE outperforms the "Blind" and "Refine" baselines.

	NTL				ICL			
	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)
abalone	91.4±1.7 (-2.4)	93.3±1.7 (-0.5)	93.4±1.0 (-0.4)	**94.6±1.4 (+0.8)**	83.1±1.5 (-10.1)	91.2±0.8 (-2.0)	93.5±1.0 (+0.3)	**93.6±0.8 (+0.4)**
annthyroid	66.1±2.8 (-19.1)	78.2±6.6 (-7.0)	83.9±7.0 (-1.3)	**85.9±4.8 (+0.7)**	65.5±2.3 (-8.7)	73.1±2.5 (-1.1)	**82.4±5.6 (+8.2)**	76.7±6.8 (+2.5)
arrhythmia	80.5±1.1 (-0.7)	82.5±0.8 (+1.3)	82.7±1.8 (+1.5)	**84.8±1.7 (+3.6)**	75.5±0.3 (-2.3)	77.1±0.7 (-0.7)	**79.2±0.2 (+1.4)**	78.4±0.8 (+0.6)
breastw	89.5±2.1 (-6.8)	96.1±0.8 (-0.2)	**99.0±0.3 (+2.7)**	98.2±0.5 (+1.9)	97.1±0.8 (-1.0)	97.4±0.8 (-0.7)	98.7±0.3 (+0.6)	**98.8±0.4 (+0.7)**
cardio	63.5±3.8 (-19.7)	76.9±3.8 (-6.3)	**92.6±3.7 (+9.4)**	85.3±4.2 (+2.1)	80.0±1.4 (-10.0)	83.3±0.9 (-6.7)	**91.1±1.9 (+1.1)**	87.5±2.1 (-2.5)
ecoli	74.9±8.2 (-24.9)	99.6±0.5 (-0.2)	**100±0.0 (+0.2)**	**100±0.0 (+0.2)**	80.4±4.2 (-8.8)	85.8±1.5 (-3.4)	88.5±1.8 (-0.7)	**89.1±0.8 (-0.1)**
forest cover	91.2±2.2 (-7.4)	**98.6±0.7 (+0.0)**	97.7±2.7 (-0.9)	**98.6±2.1 (+0.0)**	73.0±11.7 (-22.3)	77.8±6.7 (-17.5)	78.9±3.2 (-16.4)	**81.7±2.7 (-13.6)**
glass	75.1±4.0 (+2.6)	76.6±3.3 (+4.1)	**77.8±4.8 (+5.3)**	77.1±4.6 (+4.6)	54.7±11.4 (-25.9)	66.6±5.7 (-14.0)	65.4±12.0 (-15.2)	**71.5±9.2 (-9.1)**
ionosphere	95.6±0.8 (-2.3)	**96.8±0.8 (-1.1)**	96.1±1.0 (-1.8)	**96.8±0.9 (-1.1)**	92.6±1.1 (-4.9)	93.3±1.3 (-4.2)	88.7±3.3 (-8.8)	**93.4±1.0 (-4.1)**
kdd	**99.7±0.0 (-0.2)**	99.4±0.2 (-0.5)	**99.7±0.0 (-0.2)**	**99.7±0.0 (-0.2)**	**99.9±0.0 (+0.0)**	**99.9±0.0 (+0.0)**	**99.9±0.0 (+0.0)**	**99.9±0.0 (+0.0)**
kddrev	**99.5±0.1 (+0.0)**	99.4±0.1 (-0.1)	96.1±0.9 (-3.4)	**99.5±0.1 (+0.0)**	99.5±0.2 (-0.3)	99.7±0.1 (-0.1)	**99.8±0.0 (+0.0)**	99.6±0.1 (-0.2)
letter	79.8±0.5 (-5.0)	83.5±0.8 (-1.3)	76.2±6.0 (-8.6)	**84.3±4.8 (-0.5)**	82.3±2.9 (-5.4)	84.1±2.0 (-3.6)	**86.2±2.8 (-1.5)**	83.7±2.0 (-4.0)
lympho	90.8±6.7 (-6.3)	93.7±3.2 (-3.4)	96.6±1.7 (-0.5)	**98.1±2.2 (+1.0)**	94.1±2.0 (-5.3)	96.1±1.0 (-3.3)	**98.9±1.0 (-0.5)**	**98.9±1.1 (-0.5)**
mammogra.	68.7±6.2 (-13.8)	67.8±2.0 (-14.7)	69.2±3.8 (-13.3)	**78.5±3.2 (-4.0)**	64.2±4.3 (-14.8)	69.7±4.7 (-9.3)	80.0±7.7 (+1.0)	**84.0±4.3 (+5.0)**
mnist tabular	96.1±0.2 (-1.9)	**96.7±0.4 (-1.3)**	94.7±0.5 (-3.3)	96.1±0.4 (-1.9)	94.1±0.4 (-3.1)	96.4±0.3 (-0.8)	**97.9±0.1 (+0.7)**	96.3±0.2 (-0.9)
mulcross	81.7±7.5 (-17.9)	**91.2±1.4 (-8.4)**	90.8±4.5 (-8.8)	82.6±10.5 (-17.0)	93.7±4.4 (-6.3)	99.4±0.7 (-0.6)	**100±0.0 (+0.0)**	**100±0.0 (+0.0)**
musk	76.2±2.3 (-23.8)	**100±0.0 (+0.0)**	**100±0.0 (+0.0)**	**100±0.0 (+0.0)**	78.8±2.9 (-21.2)	**100±0.0 (+0.0)**	**100±0.0 (+0.0)**	**100±0.0 (+0.0)**
optdigits	31.0±3.7 (-53.7)	38.7±3.8 (-46.0)	70.9±27.8 (-13.8)	**72.6±33.6 (-12.1)**	13.8±4.2 (-83.6)	**16.3±4.3 (-81.1)**	15.9±5.1 (-81.5)	14.6±3.7 (-82.8)
pendigits	64.0±9.3 (-33.1)	85.9±6.6 (-11.2)	**99.1±0.5 (+2.0)**	98.9±0.4 (+1.8)	77.9±6.8 (-21.3)	83.3±4.7 (-15.9)	99.2±0.6 (+0.0)	**99.7±0.1 (+0.5)**
pima	59.5±3.4 (-2.2)	60.6±2.6 (-1.1)	**60.8±1.8 (-0.9)**	**60.8±1.0 (-0.9)**	58.2±3.7 (-2.1)	59.0±1.4 (-1.3)	**64.1±1.5 (+3.8)**	61.1±1.4 (+0.8)
satellite	80.9±0.4 (-1.5)	82.2±0.3 (-0.2)	82.6±0.4 (+0.2)	**82.9±0.3 (+0.5)**	78.5±1.2 (-6.7)	78.3±1.0 (-6.9)	79.3±0.9 (-5.9)	**79.5±1.0 (-5.7)**
satimage	92.3±2.1 (-7.5)	**99.7±0.1 (-0.1)**	**99.7±0.1 (-0.1)**	**99.7±0.1 (-0.1)**	89.8±1.6 (-9.9)	99.6±0.2 (-0.1)	**99.7±0.1 (+0.0)**	**99.7±0.1 (+0.0)**
seismic	51.6±0.5 (-1.3)	49.7±2.0 (-3.2)	50.3±3.0 (-2.6)	**55.6±3.8 (+2.7)**	56.9±2.7 (-6.5)	58.4±2.3 (-5.0)	**68.0±1.9 (+4.6)**	66.3±1.6 (+2.9)
shuttle	99.7±0.1 (+0.1)	**99.8±0.1 (+0.2)**	99.7±0.1 (+0.1)	99.7±0.1 (+0.1)	**99.7±0.1 (-0.3)**	99.6±0.0 (-0.4)	**99.7±0.0 (-0.3)**	**99.7±0.1 (-0.3)**
speech	48.6±1.2 (-13.9)	53.2±1.4 (-9.3)	78.8±3.0 (+16.3)	**85.5±1.6 (+23.0)**	17.1±1.9 (-41.3)	21.8±1.5 (-36.6)	**24.2±1.3 (-34.2)**	18.0±1.9 (-40.4)
thyroid	94.3±1.2 (-3.9)	96.4±0.3 (-1.8)	99.1±0.2 (+0.9)	**99.3±0.2 (+1.1)**	96.0±0.9 (-2.4)	97.7±0.3 (-0.7)	**99.4±0.2 (+1.0)**	99.2±0.3 (+0.8)
vertebral	54.8±4.6 (-5.0)	55.3±4.3 (-4.5)	47.9±12.0 (-11.9)	**59.2±9.8 (-0.6)**	43.3±1.5 (-10.5)	**50.5±2.7 (-3.3)**	45.6±5.7 (-8.2)	46.8±4.9 (-7.0)
vowels	87.6±2.2 (-10.4)	92.6±3.5 (-5.4)	**96.3±1.9 (-1.7)**	92.7±2.7 (-5.3)	91.0±2.6 (-7.9)	95.6±2.0 (-3.3)	**99.2±0.3 (+0.3)**	98.3±0.6 (-0.6)
wbc	81.2±7.0 (-11.6)	88.5±5.0 (-4.3)	**94.9±2.2 (+2.1)**	93.4±2.4 (+0.6)	86.3±2.0 (-4.6)	86.8±1.1 (-4.1)	**91.5±1.1 (+0.6)**	91.0±0.5 (+0.1)
wine	64.3±14.4 (-35.4)	93.1±7.7 (-6.6)	99.6±0.1 (-0.1)	**99.8±0.1 (+0.1)**	49.9±12.6 (-48.6)	54.6±8.3 (-43.9)	99.7±0.7 (+1.2)	**100±0.0 (+1.5)**

Appendix C

Chapter 4

C.1 Theorem 1

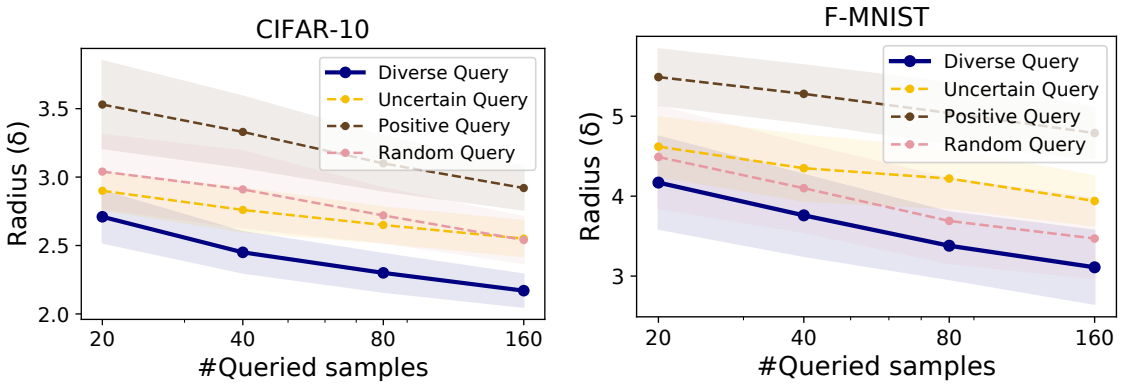


Figure C.1: Cover radius δ (Equation (C.1)) resulted from different querying strategies on the first class of CIFAR-10 and F-MNIST. Diverse queries systematically have smaller cover radius than other querying strategies.

Proof. Since S is λ_s-Lipschitz continuous and $\mathbf{u}_a$ and $\mathbf{u}_n$ are assumed to be closer than δ to $\mathbf{x}_a$ and $\mathbf{x}_n$ respectively, we have $S(\mathbf{x}_a) - \delta\lambda_s \leq S(\mathbf{u}_a)$ and $-S(\mathbf{x}_n) - \delta\lambda_s \leq -S(\mathbf{u}_n)$. Adding the inequalities and using the condition $S(\mathbf{x}_a) - S(\mathbf{x}_n) \geq 2\delta\lambda_s$, yields $0 \leq S(\mathbf{x}_a) - S(\mathbf{x}_n) - 2\delta\lambda_s \leq S(\mathbf{u}_a) - S(\mathbf{u}_n)$, which proves $S(\mathbf{u}_a) \geq S(\mathbf{u}_n)$. □

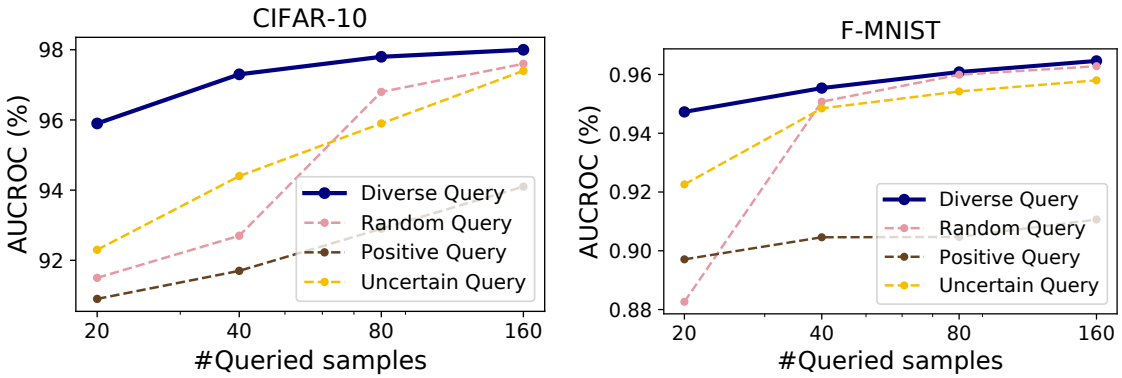


Figure C.2: Ranking performance of unlabeled data. AUC of unqueried data is evaluated using the fitted anomaly detector on the queried data. Our proposed diverse querying (k-means++) provides better ranking of the unlabeled data.

In Theorem 4.1, we considered using the fixed-radius neighborhood (δ-ball) of the queried data as the cover of the whole dataset, and mentioned diverse querying has a smaller radius than other querying strategies. In this section, we will empirically verify this fact and further illustrate diverse querying leads to good ranking of un-queried data (see also Figure C.7 on test data).

As defined in Theorem 4.1, the radius is the smallest distance that is required for any un-queried sample to be covered by the neighborhood of a queried sample of the same type. Mathematically, we compute the radius as

$$\delta = \max_{i \in \mathcal{U}} \min_{j \in \mathcal{Q}, y_i = y_j} d(\mathbf{x}_i, \mathbf{x}_j), \tag{C.1}$$

where we adopt the euclidean distance in the feature space for a meaningful metric d. We apply NTL on the first class of CIFAR-10 and F-MNIST dataset. We make queries with different budgets, after which we compute δ by Equation (C.1). We repeat this procedure for 100 times and report the mean and standard deviation in Figure C.1. We compared four querying strategies: diverse queries (k-means++), uncertain queries (Mar), positive queries (Pos1), and random queries (Rand1). It shows that diverse queries significantly lead the smallest radius δ among the compared strategies on all querying budgets.

Next, we provide an empirical, overall justification of Theorem 4.1 (see also Figure C.7 on test data). An implication of Theorem 4.1 is that, assuming anomaly scores are fixed, a smaller δ will satisfy the large anomaly score margin $(S(\mathbf{x}_a) - S(\mathbf{x}_n))$ more easily, hence it is easier for S to correctly rank the remaining unlabeled points. To justify this implication, we need a metric of ranking. AUC satisfies this requirement as it is alternatively defined as [Mohri et al., 2018, 10.5.2][Cortes and Mohri, 2003]

$$\text{AUC} = \frac{1}{|\mathcal{U}_0| + |\mathcal{U}_1|} \sum_{n \in \mathcal{U}_0, a \in \mathcal{U}_1} \mathbb{1}(S(\mathbf{u}_a) > S(\mathbf{u}_n)) \approx P_{n \in \mathcal{U}_0, a \in \mathcal{U}_1}(S(\mathbf{u}_a) > S(\mathbf{u}_n))$$

which measures the probability of ranking unlabeled samples $\mathbf{u}_a$ higher than $\mathbf{u}_n$ in terms of their scores. $\mathcal{U} = \mathcal{U}_0 \bigcup \mathcal{U}_1$ is the un-queried data indices and $\mathcal{U}_0$ and $\mathcal{U}_1$ are disjoint un-queried normal and abnormal data sets respectively. $\mathbf{u}_a$ and $\mathbf{u}_n$ are instances of each kind. We conducted experiments on CIFAR-10 and F-MNIST, where we trained an anomaly detector (NTL) on the queried data for 30 epochs and then compute the AUC on the remaining un-queried data. The results of four querying straties are reported in Figure C.2, which shows that our proposed diverse querying strategy generalizes the anomaly score ranking the best to the unqueried data among the compared strategies, testifying our analysis in the main paper. A consequence is that diverse querying can provide accurate assignments of the latent anomaly labels, which will further help learn a high-quality of anomaly detector through the unsupervised loss term in Equation (4.3).

Optimality of Cover Radius. Although k-means++ greedily samples the queries which may have a sub-optimal cover radius, greedy sampling strategies for selecting a diverse set of datapoints in a multi-dimensional space are known to produce nearly optimal solutions [Krause and Golovin, 2014], with significant runtime savings over more sophisticated search methods. As a results, we follow common practice (e.g. Arthur and Vassilvitskii [2007]) and also use the greedy approach. We check the diversity of the rustling query set by comparing

all sampling strategies considered in the paper in terms of data coverage. Figure 4 shows that the greedy strategy we use achieves the best coverage, i.e. results in the most diverse query set.

On the Assumptions of Theorem 4.1. In the proof, we assume a Lipschitz continuous S and a large margin between $S(\mathbf{x}_a)$ and $S(\mathbf{x}_n)$. Lipschitz continuity serves as a working assumption and is a common assumption when analyzing optimization landscapes of deep learning. Lipschitz continuity can be controlled by the strength of regularization on the model parameters. The large margin condition is achieved by optimizing our loss function. The supervised anomaly detection loss encourages a large margin as it minimizes the anomaly score of queried normal data and maximizes the score of the queried abnormal data. If the anomaly score function doesn't do well for the queried samples, then it should be optimized further. Our empirical results also show this is a reasonable condition.

C.2 Theorem 2

In this section, we will empirically justify the assumptions we made in Section 4.3.6 that are used to build an unbiased estimator of the anomaly ratio α (Equation (4.4)). We will also demonstrate the robustness of the estimation under varying α.

C.2.1 Proof

Proof. Let A1 and A2 denote Assumption 1 and 2, respectively. Furthermore, let $q(\mathbf{x}_1, ..., \mathbf{x}_{|\mathcal{Q}|})$ and $q_s(s_1, ..., s_{|\mathcal{Q}|})$ denote the query distribution in the data and anomaly score spaces, respectively. A2 assumes $y_s(s) := y_s(S(\mathbf{x})) = y(\mathbf{x})$ for all $\mathbf{x}$. So the expectation of Equation (4.4)

is

$$\mathbb{E}[\hat{\alpha}] = \mathbb{E}_{q(\mathbf{x}_1,\ldots,\mathbf{x}_{|Q|})}\left[\frac{1}{|\mathcal{Q}|}\sum_{i=1}^{|\mathcal{Q}|}\frac{p_s(S(\mathbf{x}_i))}{q_s(S(\mathbf{x}_i))}y(\mathbf{x})\right] \overset{A2}{=} \mathbb{E}_{q_s(s_1,\ldots,s_{|Q|})}\left[\frac{1}{|\mathcal{Q}|}\sum_{i=1}^{|\mathcal{Q}|}\frac{p_s(s_i)}{q_s(s_i)}y_s(s_i)\right]$$

$$\overset{A1}{=} \mathbb{E}_{\prod_{i=1}^{|Q|} q_s(s_i)}\left[\frac{1}{|\mathcal{Q}|}\sum_{i=1}^{|\mathcal{Q}|}\frac{p_s(s_i)}{q_s(s_i)}y_s(s_i)\right] = \frac{1}{|\mathcal{Q}|}\sum_{i=1}^{|\mathcal{Q}|}\mathbb{E}_{q_s(s_i)}\left[\frac{p_s(s_i)}{q_s(s_i)}y_s(s_i)\right] = \mathbb{E}_{p_s(s)}[y_s(s)]$$

$$= \mathbb{E}_{p(\mathbf{x})}[y_s(S(\mathbf{x}))] \overset{A2}{=} \mathbb{E}_{p(\mathbf{x})}[y(\mathbf{x})] = \alpha$$

where the change of variables makes necessary assumptions, including the existence of density functions. □

C.2.2 Assumption 1

We verify Assumption 1 by showing the correlation matrix in Figure C.3, where we jointly queried 20 points with diversified querying strategy and repeated 1000 times on two classes of CIFAR-10 and F-MNIST. Then the correlation between each pair of points are computed and placed in the off-diagonal entries. For each matrix, we show the average, maximum, and minimum of the off-diagonal terms

- CIFAR-10 Class 1: -0.001, 0.103, -0.086
- CIFAR-10 Class 2: -0.001, 0.085, -0.094
- F-MNIST Class 1: -0.001, 0.081, -0.075
- F-MNIST Class 2: -0.005, 0.087, -0.067

Which shows the correlations $\langle S(\mathbf{x}_i), S(\mathbf{x}_j)\rangle$ are negligible, and the anomaly scores can be considered approximately independent random variables.

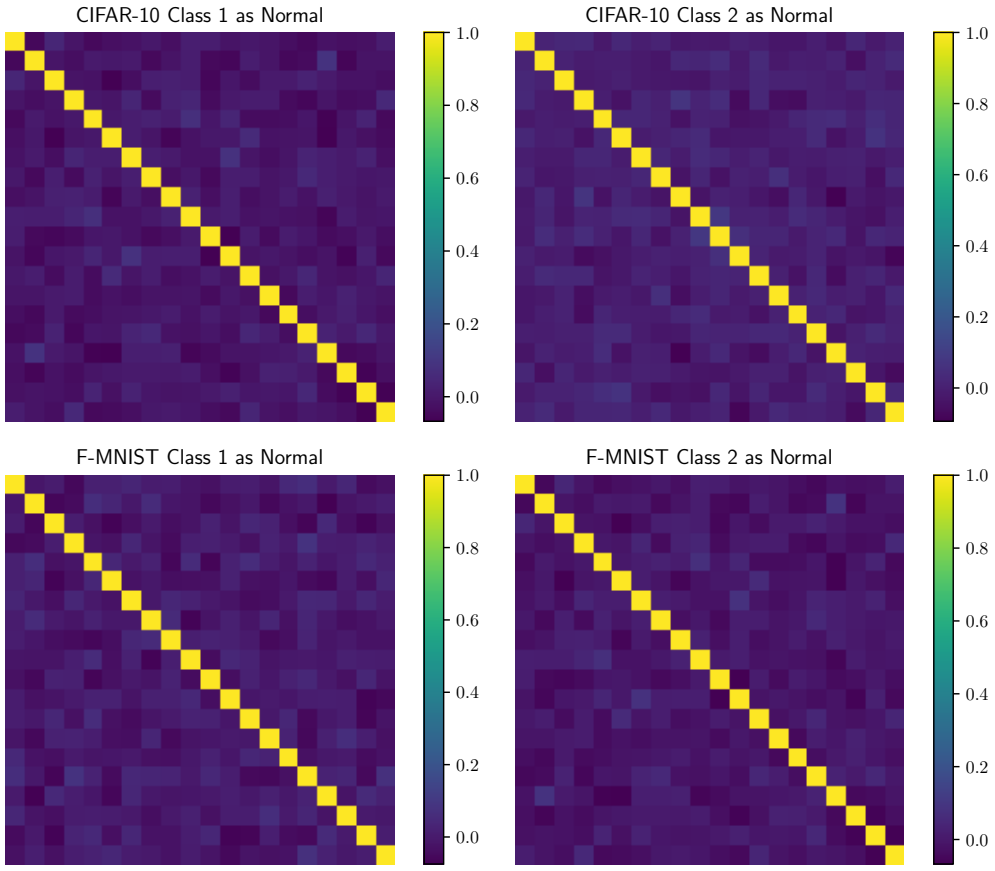


Figure C.3: Anomaly score correlation matrix $\langle S(\mathbf{x}_i), S(\mathbf{x}_j)\rangle$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are jointly sampled in the same query set. The result indicates that anomaly scores can be considered as approximately independent random variables.

C.2.3 Assumption 2

We verify Assumption 2 by counting the violations, i.e., $S(\mathbf{x}_i) = S(\mathbf{x}_j)$ but $y(\mathbf{x}_i) \neq y(\mathbf{x}_j)$ (because Assumption 2 states $y_s(s_i) = y(\mathbf{x}_i)$ and $y_s(s_j) = y(\mathbf{x}_j)$, $S(\mathbf{x}_i) = S(\mathbf{x}_j)$ implies $y(\mathbf{x}_i) = y_s(s_i) = y_s(s_j) = y(\mathbf{x}_j)$. The negation is $S(\mathbf{x}_i) = S(\mathbf{x}_j)$ and $y(\mathbf{x}_i) \neq y(\mathbf{x}_j)$.). We run the experiments on both CIFAR-10 and FMNIST. We apply the "one-vs.-rest" setup for both datasets and set the first class as normal and all the other classes as abnormal. We set the ground-truth anomaly ratio as 0.1. After the initial training, we count the pairs of data points that satisfy $S(\mathbf{x}_i) = S(\mathbf{x}_j)$ but $y(\mathbf{x}_i) \neq y(\mathbf{x}_j)$ for $i \neq j$. Our validation shows that on FMNIST, among 6666 training data points, there are 38 pairs of matching scores, and

none of them have opposite labels, and on CIFAR-10, among 5555 training data points, the numbers are 21 and 3, respectively.

C.2.4 Contamination Ratio Estimation

Table C.1: Estimated contamination ratios on CIFAR-10 and F-MNIST when $|\mathcal{Q}| = 40$ and the backbone model is NTL. The first row shows the true contamination ratio ranging from 1% to 45%. The estimations are repeated 50 times.

	1%	5%	10%	15%	20%
CIFAR-10	$0.5\% \pm 1.2\%$	$6.0\% \pm 3.3\%$	$12.0\% \pm 4.4\%$	$15.3\% \pm 4.5\%$	$18.9\% \pm 5.4\%$
F-MNIST	$1.0\% \pm 1.5\%$	$3.8\% \pm 2.3\%$	$8.7\% \pm 4.1\%$	$12.8\% \pm 5.3\%$	$19.3\% \pm 5.1\%$
	25%	30%	35%	40%	45%
CIFAR-10	$26.2\% \pm 6.0\%$	$30.6\% \pm 5.5\%$	$35.8\% \pm 6.9\%$	$42.0\% \pm 7.7\%$	$47.2\% \pm 6.7\%$
F-MNIST	$27.9\% \pm 6.4\%$	$31.8\% \pm 6.1\%$	$38.3\% \pm 6.5\%$	$43.1\% \pm 5.7\%$	$48.9\% \pm 5.6\%$

We estimate the contamination ratio by Equation (4.4) under varying true ratios. This part shows the estimated contamination ratio when the query budget is $|\mathcal{Q}| = 40$. The estimations from the backbone model NTL is shown in Table C.1. The first row contains the ground truth contamination rate, and the second and third row indicate the inferred values for two datasets, using our approach. Most estimates are withing the error bars and hence accurate. The estimation errors for low ground-truth contamination ratios are acceptable as confirmed by the sensitivity study in [Qiu et al., 2022b] which concludes that the LOE approach still works well if the anomaly ratio is mis-specified within 5 percentage points. Interestingly, we find the estimation error increases somewhat with the contamination ratio. However, a contamination ratio larger than 40% is rare in practice (most datasets should be fairly clean and would otherwise require additional preprocessing). In an anomaly detection benchmark (`https://github.com/Minqi824/ADBench`), none of the datasets have an anomaly ratio larger than 40%.

C.3 Baselines Details

In this section, we describe the details of the baselines in Table 4.1 in the main paper. For each baseline method, we explain their query strategies and post-query training strategies we implement in our experiment. Please also refer to our codebase for practical implementation details.

- **Rand1.** This strategy used by Ruff et al. [2019] 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,

$$\mathcal{L}_{\text{Rand1}}(\theta) = \frac{1}{|\mathcal{Q}|} \sum_{j \in \mathcal{Q}} \left(y_j \mathcal{L}_a^\theta(\mathbf{x}_j) + (1 - y_j) \mathcal{L}_n^\theta(\mathbf{x}_j) \right) + \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \mathcal{L}_n^\theta(\mathbf{x}_i). \tag{C.2}$$

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

- **Rand2.** The querying strategy of Trittenbach et al. [2021] 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 $\mathcal{L}_{\text{Rand1}}(\theta)$ (Equation (C.2)).

- **Mar.** After training the backbone model for one epoch, this querying strategy by Görnitz et al. [2013] uses the α-quantile (s_α) of the training data anomaly scores to define a "normality region". Then the $|\mathcal{Q}|$ 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 $\mathcal{L}_{\text{Rand1}}(\theta)$ (Equation (C.2)). Note that in practice we don't know the true anomaly ratio for the α-quantile. In all experiment, 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 SOEL.

- **Hybr1.** This hybrid strategy, also used by [Görnitz et al., 2013] 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

$$\arg\min_{1\leq i\leq N} 0.5 + \frac{|\{j \in \mathrm{NN}_k(\phi(\mathbf{x}_i)) : j \in \mathcal{Q}\}|}{2k} + \beta \frac{\|s_i - s_\alpha\| - \min_i \|s_i - s_\alpha\|}{\max_i \|s_i - s_\alpha\| - \min_i \|s_i - s_\alpha\|}$$

  where the inter-sample distance is measured in the feature space and the number of nearest neighbors is $k = \lceil N/|\mathcal{Q}| \rceil$. We set $\beta = 1$ for equal contribution of both terms. After the queries are labeled, the training loss function is the same as $\mathcal{L}_{\mathrm{Rand1}}(\theta)$ (Equation (C.2)).

- **Pos1.** This querying strategy by Pimentel et al. [2020] 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

$$\mathcal{L}_{\mathrm{Pos1}}(\theta) = \frac{1}{|\mathcal{Q}|} \sum_{j\in\mathcal{Q}} \big(y_j \mathcal{L}_a^\theta(\mathbf{x}_j) + (1 - y_j)\mathcal{L}_n^\theta(\mathbf{x}_j)\big).$$

  Pimentel et al. [2020] 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 [Hendrycks et al., 2018, Ruff et al., 2019].

- **Pos2.** This approach of [Barnabé-Lortie et al., 2015] 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 $\mathcal{L}_{\mathrm{Rand1}}(\theta)$ (Equation (C.2)).

- **Hybr2.** This hybrid strategy by Das et al. [2019] makes positive diverse queries. It combines querying according to anomaly scores with distance-based diversification. Hybr2 se-

lects 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

$$\arg\max_{1\leq i\leq N} \frac{s_i - \min_i s_i}{\max_i s_i - \min_i s_i} + \beta \min_{j\in\mathcal{Q}} \frac{d(\mathbf{x}_i, \mathbf{x}_j) - \min_{a\neq b} d(\mathbf{x}_a, \mathbf{x}_b)}{\max_{a\neq b} d(\mathbf{x}_a, \mathbf{x}_b) - \min_{a\neq b} d(\mathbf{x}_a, \mathbf{x}_b)}$$

where $d(\mathbf{x}_i, \mathbf{x}_j) = ||\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)||_2$. We set $\beta = 1$ for equal contribution of both terms. After the queries are labeled, Das et al. [2019] 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

$$\mathcal{L}_{\text{Hybr2}}(\theta) = \frac{1}{|\mathcal{Q}|} \sum_{j\in\mathcal{Q}} \big(y_j \mathcal{L}_a^\theta(\mathbf{x}_j) + (1-y_j)\mathcal{L}_n^\theta(\mathbf{x}_j)\big).$$

- **Hybr3.** This baseline by [Ning et al., 2022] uses the same query strategy as Hybr2, but differs in the training loss function,

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

where $w_j = 2\sigma(d_j)$ and $\hat{w}_i = 2 - 2\sigma(d_i)$ where $\sigma(\cdot)$ is the Sigmoid function and $d_i = 10c_d\big(||\phi(\mathbf{x}_i) - \boldsymbol{c}_0||_2 - ||\phi(\mathbf{x}_i) - \boldsymbol{c}_1||_2\big)$ 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 c_d is the min-max normalization factor.

We make three observations for the loss function. First, $\mathcal{L}_{\text{Hybr3}}(\theta)$ filters out all labeled anomalies in the supervised learning part and puts a large weight (but only as large as 2 at most) to the true normal data that has a high anomaly score. Second, in the unlabeled data, $\mathcal{L}_{\text{Hybr3}}(\theta)$ puts 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 SOEL, which weighs labeled data $|\mathcal{U}|/|\mathcal{Q}|$ times higher than unlabeled data.

Algorithm 2: Training Procedure of SOEL

Input: Unlabeled training dataset $\mathcal{D}$, querying budget K
Procedure:
Train the model on $\mathcal{D}$ for one epoch as if all data were normal;
Query K data points from $\mathcal{D}$ diversely resulting in a labeled set $\mathcal{Q}$ and an unlabeled set $\mathcal{U}$;
Estimate the contamination ratio α based on $\mathcal{Q}$;
Finally train the model with $\{\mathcal{Q},\mathcal{U}\}$ until convergence:
For each iteration:
We construct a mini-batch with $\mathcal{Q}$ and a subsampled mini-batch of $\mathcal{U}$
The sample in $\mathcal{Q}$ is up-weighted with $1/|\mathcal{Q}|$ and the sample in $\mathcal{U}$ is down-weighted with weight $1/|\mathcal{U}|$
The training strategy for $\mathcal{Q}$ is supervised learning; the training strategy for $\mathcal{U}$ is LOE with the estimated anomaly ratio α.

C.4 Implementation Details

In this section, we present the implementation details in the experiments. They include an overall description of the experimental procedure for all datasets, model architecture, data split, and details about the optimization algorithm.

C.4.1 Experimental Procedure

We apply the same experimental procedure for each dataset and each compared method. The experiment starts with an unlabeled, contaminated training dataset with index set $\mathcal{U}$. We first train the anomaly detector on $\mathcal{U}$ for one epoch as if all data were normal. Then we conduct the diverse active queries at once and estimate the contamination ratio α by the importance sampling estimator Equation (4.4). Lastly, we optimize the post-query training losses until convergence. The obtained anomaly detectors are evaluated on a held-out test set. The training procedure of SOEL is shown in Algorithm 2.

C.4.2 Data Split

Image Data. For the image data including both natural (CIFAR-10 [Krizhevsky et al., 2009] and F-MNIST [Xiao et al., 2017]) and medical (MedMNIST [Yang et al., 2021b]) images, we use the original training, validation (if any), and test split. When contaminating the training data of one class, we randomly sample images from other classes' training data and leave the validation and test set untouched. Specifically for DermaMNIST in MedMNIST, we only consider the classes that have more than 500 images in the training data as normal data candidates, which include benign keratosis-like lesions, melanoma, and melanocytic nevi. We view all other classes as abnormal data. Different experiment runs have different randomness.

Tabular Data. Our study includes the four multi-dimensional tabular datasets from the ODDS repository[1] which have an outlier ratio of at least 30%. . 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. Different experiment runs have different randomness.

Video Data. We use UCSD Peds1[2], a benchmark dataset for video anomaly detection. UCSD Peds1 contains 70 surveillance video clips – 34 training clips and 36 testing clips. Each frame is labeled to be abnormal if it has non-pedestrian objects and labeled normal otherwise. Making the same assumption as [Pang et al., 2020], we treat each frame independent and mix the original training and testing clips together. This results in a dataset of 9955 normal frames and 4045 abnormal frames. We then randomly sub-sample 6800 frames out of the

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

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

normal frames and 2914 frames out of the abnormal frames without replacement to form a contaminated training dataset with 30% anomaly ratio. A same ratio is also used in the literature [Pang et al., 2020] that uses this dataset. The remaining data after sampling is used for the testing set, whose about 30% data is anomalous. Like the other data types, different experiment runs have different randomness for the training dataset construction.

C.4.3 Model Architecture

The experiments involve two anomaly detectors, NTL and MHRot, and three data types.

NTL on Image Data and Video Data. For all images (either natural or medical) and video frames, we extract their features by feeding them into a ResNet152 pre-trained on ImageNet and taking the penultimate layer output for our usage. The features are kept fixed during training. We then train an NTL on those features. We apply the same number of transformations, network components, and anomaly loss function $\mathcal{L}_a^\theta(\mathbf{x})$, as when Qiu et al. [2022b] apply NTL on the image data.

NTL on Tabular Data. We directly use the tabular data as the input of NTL. We apply the same number of transformations, network components, and anomaly loss function $\mathcal{L}_a^\theta(\mathbf{x})$, as when Qiu et al. [2022b] apply NTL on the tabular data.

MHRot on Image Data. We use the raw images as input for MHRot. We set the same transformations, MHRot architecture, and anomaly loss function as when Qiu et al. [2022b] apply MHRot on the image data.

DSVDD on Image Data. For all images (either natural or medical), we build DSVDD on the features from the penultimate layer of a ResNet152 pre-trained on ImageNet. The features are kept fixed during training. The neural network of DSVDD is a three-layer MLP with intermediate batch normalization layers and ReLU activation. The hidden sizes are $[1024, 512, 128]$.

C.4.4 Optimization Algorithm

Model	Dataset	Learning Rate	Epoch	Minibatch Size	τ
NTL	CIFAR-10	1e-4	30	512	1e-2
	F-MNIST	1e-4	30	512	1e-2
	MedMNIST	1e-4	30	512	1e-2
	ODDS	1e-3	100	$\lceil N/5 \rceil$	1e-2
	UCSD Peds1	1e-4	3*	192	1e-2
MHRot	CIFAR-10	1e-3	15	10	N/A
	F-MNIST	1e-4	15**	10	N/A
	MedMNIST	1e-4	15	10	N/A
Deep SVDD	CIFAR-10	1e-4	30	512	1e-2
	F-MNIST	1e-4	30	512	1e-2
	MedMNIST	1e-4	30	512	1e-2

*Hybr2, Hybr3, Pos1, and Pos2 train 30 epochs. All other methods train 3 epochs.
**SOEL train 3 epochs.

Table C.2: A summary of optimization parameters for all methods.

In the experiments, we use Adam [Kingma and Ba, 2015] to optimize the objective function 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, 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. For video data and tabular data, the optimization hyperparameters are set as recommended by Qiu et al. [2022b]. In order to choose τ (in Equation (4.2)), we constructed a validation dataset of CIFAR-10 to select the

parameter τ among {1, 1e-1, 1e-2, 1e-3} and applied the validated τ (1e-2) on all the other datasets in our experiments. Specifically, we split the original CIFAR-10 training data into a training set and a validation set. After validation, we train the model on the original training set again. We summarize all optimization hyperparameters in Table C.2.

When training models with SOEL, we resort to the block coordinate descent scheme that update the model parameters θ and the pseudo labels $\tilde{\mathbf{y}}$ of unlabeled data in turn. In particular, we take the following two update steps iteratively:

- update θ by optimizing Equation (4.3) given $\tilde{\mathbf{y}}$ fixed;
- update $\tilde{\mathbf{y}}$ by sovling the constrained optimization in Section 4.3.5 given θ fixed;

Upon updating $\tilde{\mathbf{y}}$, we use the LOE$_S$ variant [Qiu et al., 2022b] for the unlabeled data. We set the pseudo labels $\tilde{\mathbf{y}}$ by performing the optimization below

$$\min_{\tilde{\mathbf{y}}\in\{0,0.5\}^{|\mathcal{U}|}} \frac{1}{|\mathcal{U}|}\sum_{i\in\mathcal{U}} \tilde{y}_i\mathcal{L}_a^\theta(\mathbf{x}_i) + (1-\tilde{y}_i)\mathcal{L}_n^\theta(\mathbf{x}_i) \qquad \text{s.t.} \qquad \sum_{i=1}^{|\mathcal{U}|}\tilde{y}_i = \frac{\tilde{\alpha}|\mathcal{U}|}{2},$$

where $\tilde{\alpha}$ is the updated contamination ratio of $\mathcal{U}$ after the querying round, $\tilde{\alpha} = \big(\alpha N - \sum_{j\in\mathcal{Q}} y(\mathbf{x}_j)\big)/|\mathcal{U}|$, and α is computed by Equation (4.4) given $\mathcal{Q}$. The solution is to rank the data by $\mathcal{L}_n^\theta(\mathbf{x}) - \mathcal{L}_a^\theta(\mathbf{x})$ and label the top $\tilde{\alpha}$ data abnormal (equivalently setting $\tilde{y} = 0.5$) and all the other data normal (equivalently $\tilde{y} = 0$).

When we compute the Euclidean distance in the feature space, we construct the feature vector of a sample by concatenating all its encoder representations of different transformations. For example, if the encoder representation has 500 dimensions and the model has 10 transformations, then the final feature representation has $10 \times 500 = 5000$ dimensions.

C.4.5 Time Complexity

Regarding the time complexity, the optimization uses stochastic gradient descent. The complexity of our querying strategy is $O(KN)$ where K is the number of queries and N is the size of the training data. This complexity can be further reduced to $O(K \log N)$ with a scalable extension of k-means++ [Bahmani et al., 2012].

C.5 Additional Experiments and Ablation Study

The goal of this ablation study is to show the generality of SOEL, to better understand the success of SOEL, and to disentangle the benefits of the training objective and the querying strategy. To this end, we applied SOEL to different backbone models and different data forms (raw input and embedding input), performed specialized experiments to compare the querying strategies, to demonstrate the optimality of the proposed weighting scheme in Equation (4.3), and to validate the detection performance of the estimated ratio by Equation (4.4). We also compared SOEL against additional baselines including semi-supervised learning frameworks and shallow anomaly detectors.

C.5.1 Randomness of Initialization

Random Initialization affects both the queried samples and downstream performance. To evaluate the effects, we ran all experiments 5 times with different random seeds and reported all results with error bars. In Figure C.1 we can see that the radius of the cover (a smaller radius means the queries are more diverse) does have some variance due to the random initialization. However, the corresponding results in terms of detection accuracy in Figure 4.2 do have very low variance. Our interpretation is that for the CIFAR10 and F-MNIST

experiments, the random initialization has little effect on detection performance.

C.5.2 Results with Other Backbone Models

Table C.3: $|\mathcal{Q}| = 20$. AUC (%) with standard deviation for anomaly detection on six datasets (CIFAR-10, F-MNIST, Blood, OrganA, OrganC, OrganS). The backbone models are MHRot [Hendrycks et al., 2019] and Deep SVDD [Ruff et al., 2018]. For all experiments, we set the contamination ratio as 10%. SOEL consistently outperforms two best-performing baselines on all six datasets.

	MHRot			Deep SVDD		
	SOEL	Hybr1	Hybr2	SOEL	Hybr1	Hybr2
CIFAR-10	**86.9±0.7**	83.9±0.1	49.1±2.0	**93.1±0.2**	89.0±0.6	91.3±1.0
F-MNIST	**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

We are interested whether SOEL works for different backbone models. To that end, we repeat part of the experiments in Table 4.2 but using an self-supervised learning model MHRot [Hendrycks et al., 2019] and a one class classification model Deep SVDD [Ruff et al., 2018] as the backbone model. We compare SOEL to two best performing baselines — Hybr1 and Hybr2. In this experiment, MHRot and Deep SVDD take different input types: while MHRot takes raw images as input, Deep SVDD uses pre-trained image features. We also set the query budget to be $|\mathcal{Q}| = 20$.

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

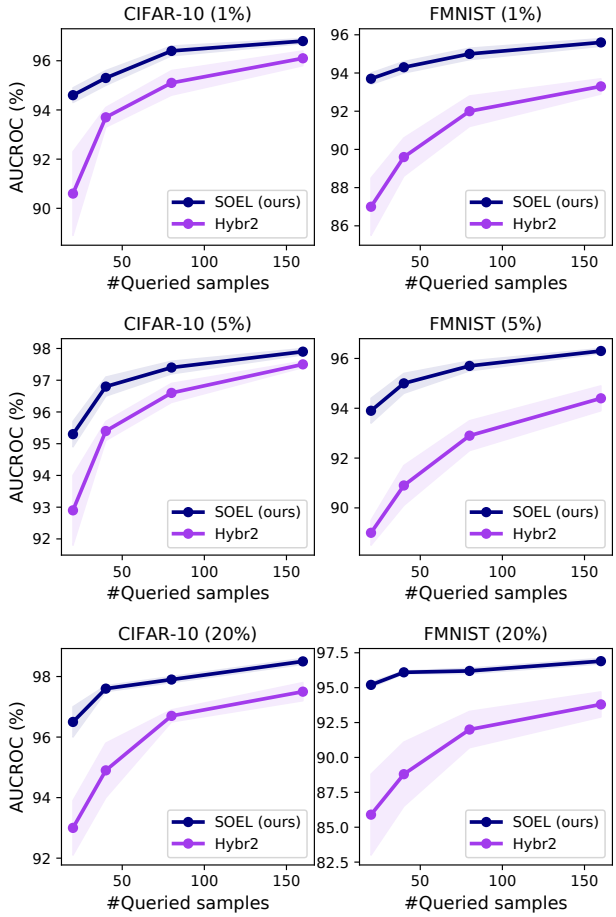


Figure C.4: Running AUCs (%) with different query budgets and data contamination ratios (1%-top row, 5%-middle row, 20%-bottom row). Models are evaluated at $20, 40, 80, 160$ queries. SOEL performs the best on all three contamination ratio setups.

C.5.3 Robustness to Anomaly Ratios

Our method works for both low anomaly ratios and high anomaly ratios. In Figure C.4, we compare SOEL against the best-performing baseline Hybr2 on CIFAR-10 and FMNIST benchmarks. We vary the anomaly ratio among 1%, 5%, and 20%. On all these three anomaly ratio settings, SOEL has significantly better performance than the baseline by over 2 percentage points on average.

Table C.4: $|\mathcal{Q}| = 20$. AUC (%) with standard deviation for anomaly detection on CIFAR-10 and F-MNIST. For all experiments, we set the contamination ratio as 10%. SOEL mitigates the performance drop when NTL and MHRot trained on the contaminated datasets. Results of the unsupervised method LOE are borrowed from Qiu et al. [2022b].

	NTL			MHRot		
	LOE	k-means++	SOEL	LOE	k-means++	SOEL
CIFAR-10	94.9±0.1	95.6±0.3	**96.3±0.3**	86.3±0.2	64.0±0.2	**86.9±0.7**
F-MNIST	92.5±0.1	94.3±0.2	**94.8±0.4**	91.2±0.4	91.5±0.1	**92.6±0.1**

C.5.4 Disentanglement of SOEL

We disentangle the benefits of each component of SOEL and compare it to unsupervised anomaly detection with latent outlier exposure (LOE) [Qiu et al., 2022b], and to supervised active anomaly detection with k-means++ querying strategy. Both active approaches (k-means++ and SOEL) are evaluated with $|\mathcal{Q}| = 20$ labeled samples. The unsupervised approach LOE requires an hyperparameter of the assumed data contamination ratio, which we set to the ground truth value 10%. Comparing SOEL to LOE reveals the benefits of the k-means++ active approach[3]; comparing SOEL to k-means++ reveals the benefits of the unsupervised loss function in SOEL. Results in Table C.4 show that SOEL leads to improvements for both ablation models.

C.5.5 Comparison to Binary Classifier

In the semi-supervised anomaly detection setup, the labeled points can be seen as an imbalanced binary classification dataset. We, therefore, perform an ablation study where we only replace deep anomaly detection backbone models with a binary classifier. All the other training and querying procedures are the same. We report the results on four different querying budget situations in Figure C.5. The figure shows that a binary classifier on all

[3]Notice that while LOE uses the true contamination ratio (an oracle information), SOEL only uses the estimated contamination ratio by the 20 queries.

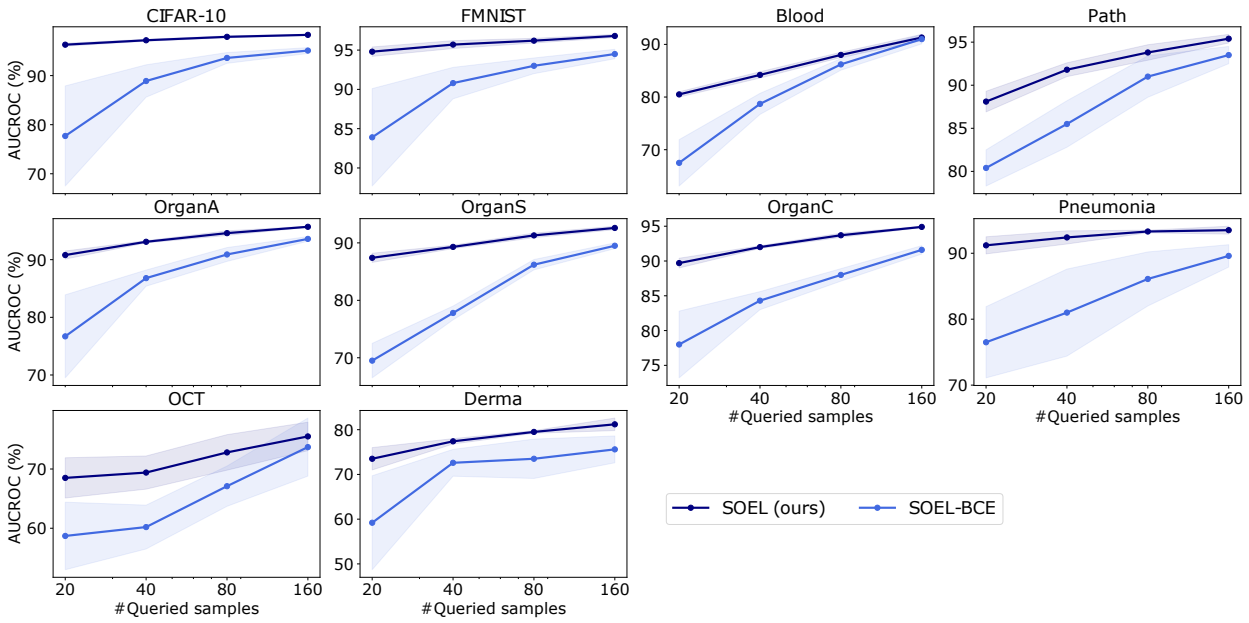


Figure C.5: Running AUCs (%) with different query budgets. Models are evaluated at $20, 40, 80, 160$ queries. Deep anomaly detection model (NTL) performs significantly better than a binary classifier.

11 image datasets falls far short of the NTL, a deep anomaly detection model. The results prove that the inductive bias (learning compact representations for normal data) used by anomaly detection models are useful for anomaly detection tasks. However, such inductive bias is lacking for binary classifiers. Especially when only querying as few as 20 points, the model can't see all anomalies. The decision boundary learned by the classifier based on the queried anomalies possibly doesn't generalize to the unseen anomalies.

C.5.6 Comparison to a Batch Sequential Setup

In Figure C.6, we extend our proposed method SOEL to a sequential batch active anomaly detection setup. This sequential extension is possible because our querying strategy k-means++ is also a sequential one. At each round, we query 20 points and update the estimated contamination ratio. We plot this sequential version of SOEL and the original SOEL in Figure C.6 and make comparisons. The sequential version is not as effective as a

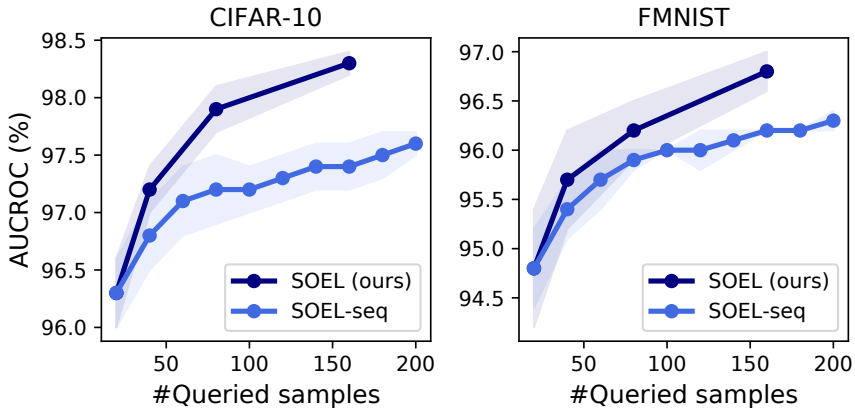


Figure C.6: Running AUCs (%) with different query budgets. Models are evaluated at $20, 40, 80, 160$ queries. SOEL performs better than a sequential version.

single batch query of SOEL.

C.5.7 Comparisons of Querying Strategies

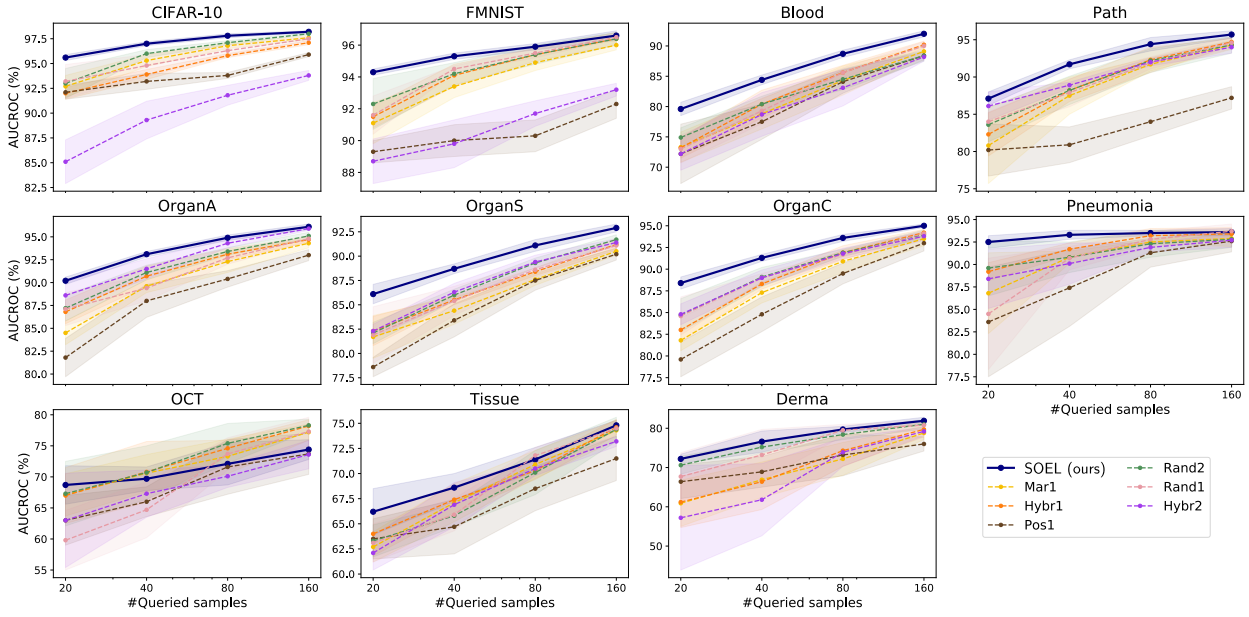


Figure C.7: Ablation study on the query strategy. K-Means++ significantly outperforms other strategies for active anomaly detection on most of the datasets.

To understand the benefit of sampling diverse queries with k-means++ and to examine the generalization ability (stated in Theorem 4.1) of different querying strategies, we run the

following experiment: We use a supervised loss on labeled samples to train various anomaly detectors. The only difference between them is the querying strategy used to select the samples. We evaluate them on all image data sets we study for varying number of queries $|\mathcal{Q}|$ between 20 and 160.

Results are in Figure C.7. On all datasets except OCT, k-means++ consistently outperforms all other querying strategies from previous work on active anomaly detection. The difference is particularly large when only few samples are queried. This also confirms that diverse querying generalizes better on the test data than other querying strategies (see additional results in Appendix C.1).

C.5.8 Ablation on Estimated Contamination Ratio

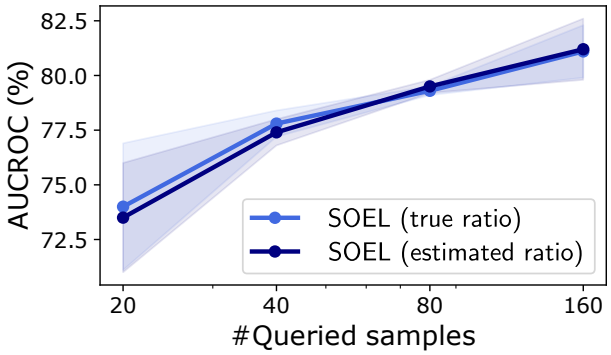


Figure C.8: Model using the estimated ratio is indistinguishable from the one using the true ratio.

To see how the estimated ratio affects the detection performance, we compare SOEL to the counterpart with the true anomaly ratio. We experiment on all 11 image datasets. In Fig. C.8, we report the average results for all datasets when querying $|\mathcal{Q}| = 20, 40, 80, 160$ samples. It shows that SOEL with either true ratio or estimated ratio performs similar given all query budgets. Therefore, the estimated ratio can be applied safely. This is very important in practice, since in many applications the true anomaly ratio is not known.

C.5.9 Ablations on Weighting Scheme

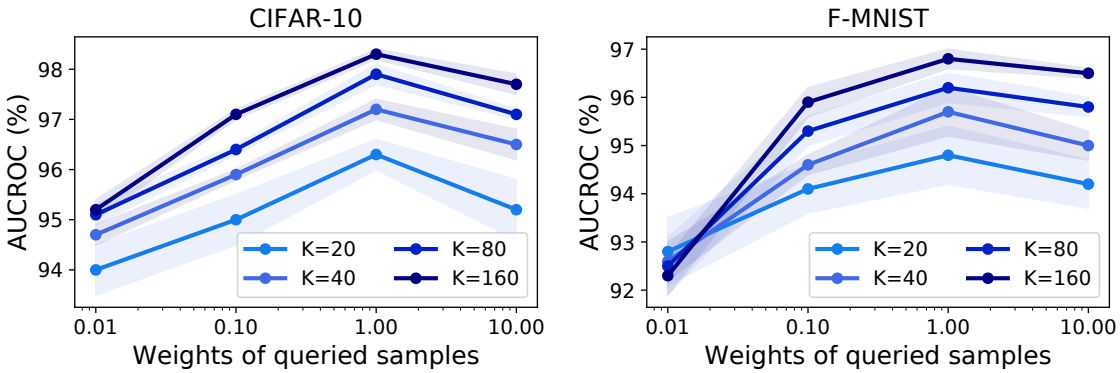


Figure C.9: Ablation study on the weighting scheme in Equation (4.3). With different query budgets $|\mathcal{Q}|$, the performance on image datasets degrades both upon down-weighting (0.01, 0.1) or up-weighting (10.0) the queried samples. In contrast, equal weighting yields optimal results.

We make the implicit assumption that the *averaged* losses over queried and unqueried data should be equally weighted (Equation (4.3)). That means, if a fraction ϵ of the data is queried, every queried data point weights $1/\epsilon$ as much as an unqueried datum. As a consequence, neither the queried nor the unqueried data points can dominate the result.

To test whether this heuristic is indeed optimal, we added a scalar prefactor to the supervised loss in Equation (4.3) (the first term) and reported the results on the CIFAR-10 and F-MNIST datasets with different query budgets (Figure C.9). A weight<1 corresponds to downweighting the queried term, while a weight>1 corresponds to upweighting it. We use the same experimental setup and backbone (NTL) as in the paper. The results are shown in Figure C.9. We see that the performance degrades both upon down-weighting (0.01, 0.1) or up-weighting (10.0) the queried samples. In contrast, equal weighting yields optimal results.

Table C.5: Performance of ablation study on τ. AUROCs (%) on CIFAR-10 and F-MNIST when $|\mathcal{Q}| = 20$, the ground-truth contamination ratio is 0.1, and the backbone model is NTL.

τ	1	0.1	0.01	0.001
CIFAR-10	93.2 ± 1.7	94.5 ± 0.8	96.3 ± 0.3	95.9 ± 0.4
F-MNIST	91.8 ± 1.4	92.7 ± 1.1	94.8 ± 0.6	94.9 ± 0.2

C.5.10 Ablations on Temperature τ

τ (in Equation (4.2)) affects the querying procedure and smaller τ makes the querying procedure more deterministic and diverse because the softmax function (in Equation (4.2)) can eventually become a maximum function. We add an ablation study on different values of τ. We did experiments under the ground truth contamination ratio being 0.1 and $|Q| = 20$. As Table C.5 shows, the smaller τ results in better AUROC results (more diverse) and smaller errors (more deterministic).

C.5.11 Ablations on Pseudo-label Values $\tilde{y}$

Table C.6: Performance of ablation study on $\tilde{y}$. AUROC (%) on CIFAR-10 and F-MNIST when $|\mathcal{Q}| = 20$, the ground-truth contamination ratio is 0.1, and the backbone model is NTL.

$\tilde{y}$	1.0	0.875	0.75	0.625	0.5	0.25
CIFAR-10	95.3 ± 0.6	95.7 ± 0.4	95.8 ± 0.4	96.0 ± 0.5	96.3 ± 0.3	94.5 ± 0.3
F-MNIST	94.5 ± 0.5	94.5 ± 0.4	94.6 ± 0.4	94.6 ± 0.3	94.8 ± 0.6	94.0 ± 0.4

Analyzing the effects of the pseudo-label values $\tilde{y}$ is an interesting ablation study. Therefore, we perform the following experiments to illustrate the influence of different $\tilde{y}$ values. We set the ground truth contamination ratio being 0.1 and $|Q| = 20$. We vary the $\tilde{y}$ from 0.25 to 1.0 and conduct experiments. For each $\tilde{y}$ value, we run 5 experiments with different random seeds and report the AUROC results with standard deviation. It shows that $\tilde{y} = 0.5$ performs the best. While the performance of CIFAR-10 degrades slightly as $\tilde{y}$ deviates from

0.5, F-MNIST is pretty robust to $\tilde{y}$. All tested $\tilde{y}$ outperform the best baseline reported in Table 4.2.

C.5.12 Comparisons with Semi-supervised Learning Frameworks

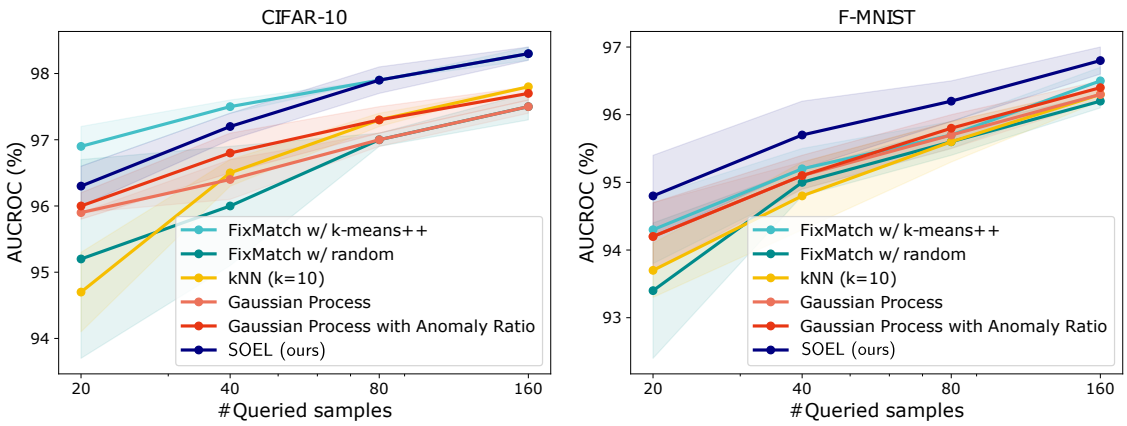


Figure C.10: Comparison with semi-supervised learning fraemworks, FixMatch [Sohn et al., 2020a], k-nearest neighbors [Iscen et al., 2019], and Gaussian process [Li et al., 2018b]. On F-MNIST, SOEL outperforms all baselines, while on CIFAR-10, SOEL has a comparable performance with FixMatch with k-means++ querying.

SOEL exploits the unlabeled data to improve the model performance. This shares the same spirit of semi-supervised learning. We are curious about how a semi-supervised learning method performs in our active anomaly detection setup. To this end, we adapted an existing semi-supervised learning framework FixMatch [Sohn et al., 2020a] to our setup and compared with our method in Figure C.10. As follows, we will first describe the experiment results and then state the adaptation of FixMatch to anomaly detection we made.

FixMatch, as a semi-supervised learning algorithm, regularizes the image classifier on a large amount of unlabeled data. The regularization, usually referred to consistency regularization, requires the classifier to have consistent predictions on different views of unlabeled data, thus improves the classifier's performance. FixMatch generates various data views through image augmentations followed by Cutout [DeVries and Taylor, 2017]. We noticed that, although

FixMatch focuses on making use of the unlabeled data, its performance is highly affected by the quality of the labeled data subset. We investigated two variants depending on how we acquire the labeled data. One is the original semi-supervised learning setting, i.e., assuming the labeled data is a random subset of the whole dataset. The other one utilizes the same diversified data querying strategy k-means++ as SOEL to acquire the labeled part. In Figure C.10, we compared the performance of the two variants with SOEL. It shows that, on natural images CIFAR10 for which FixMatch is developped, while the original FixMatch with random labeled data is still outperformed by SOEL, FixMatch with our proposed querying strategy k-means++ has a comparable performance with SOEL. However, such advantage of FixMatch diminishes for the gray image dataset F-MNIST, where both variants are beat by SOEL on all querying budgets. In addition, the FixMatch framework is restrictive and may not be applicable for tabular data and medical data, as the augmentations are specially designed for natural images.

FixMatch is designed for classification. To make it suit for anomaly detection, we adapted the original algorithm[4] and adopted the following procedure and loss function.

1. Label all training data as normal and train the anomaly detector for one epoch;
2. Actively query a subset of data with size $|\mathcal{Q}|$, resulting in $\mathcal{Q}$ and the remaining data $\mathcal{U}$;
3. Finetune the detector in a supervised manner on non-augmented $\mathcal{Q}$ for 5 epochs;
4. Train the detector with the FixMatch loss Equation (C.3) on augmented $\{\mathcal{U}, \mathcal{Q}\}$ until convergence.

We denote weak augmentation of input $\mathbf{x}$ by $\alpha(\mathbf{x})$ and the strong augmentation by $\mathcal{A}(\mathbf{x})$.

[4]We adapted the FixMatch implementation `https://github.com/kekmodel/FixMatch-pytorch`

The training objective function we used is

$$\mathcal{L}_{\text{FixMatch}}(\theta) = \frac{1}{|\mathcal{Q}|}\sum_{j\in\mathcal{Q}}\big(y_j\mathcal{L}_a^\theta(\alpha(\mathbf{x}_j)) + (1-y_j)\mathcal{L}_n^\theta(\alpha(\mathbf{x}_j))\big)$$
$$+\frac{1}{|\mathcal{U}|}\sum_{i\in\mathcal{U}}\mathbb{1}(S(\alpha(\mathbf{x}_i)) < q_{0.7} \text{ or } S(\alpha(\mathbf{x}_i)) > q_{0.05})\big(\tilde{y}_i\mathcal{L}_a^\theta(\mathcal{A}(\mathbf{x}_i)) + (1-\tilde{y}_i)\mathcal{L}_n^\theta(\mathcal{A}(\mathbf{x}_i))\big) \tag{C.3}$$

where pseudo labels $\tilde{y}_i = \mathbb{1}(S(\alpha(\mathbf{x}_i)) > q_{0.05}))$ and q_n is the n-quantile of the anomaly scores $\{S(\alpha(\mathbf{x}_i))\}_{i\in\mathcal{U}}$. In the loss function, we only use the unlabeled samples with confidently predicted pseudo labels. This is controlled by the indicator function $\mathbb{1}(S(\alpha(\mathbf{x}_i)) < q_{0.7}$ or $S(\alpha(\mathbf{x}_i)) > q_{0.05})$. We apply this loss function for mini-batches on a stochastic optimization basis.

We also extend the semi-supervised learning methods using non-parametric algorithms to our active anomaly detection framework. We applied k-nearest neighbors and Gaussian process for inferring the latent anomaly labels [Iscen et al., 2019, Li et al., 2018b] because these algorithms are unbiased in the sense that if the queried sample size is large enough, the inferred latent anomaly labels approach to the true anomaly labels. For these baselines, we also queried a few labeled data with k-means++ -based diverse querying strategy and then annotate the unqueried samples with k-nearest neighbor classifier or Gaussian process classifer trained on the queried data.

Both methods become ablations of SOEL. We compare SOEL with them on CIFAR-10 and F-MNIST under various query budgets and report their results in Figure C.10. On both datasets, SOEL improves over the variant of using only queried samples for training. On F-MNIST, SOEL outperforms all ablations clearly under all query budgets, while on CIFAR-10, SOEL outperforms all ablations except for FixMatch when query budget is low. In conclusion, SOEL boosts the performance by utilizing the unlabeled samples properly, while other labeling strategies are less effective.

C.5.13 More Comparisons

Table C.7: Comparisons with kNN method. We reported the F1-score (%) with standard error for anomaly detection on tabular datasets when the query budget $K = 10$. SOEL outperforms the kNN baseline.

	k^{th}**NN**	**ALOE**
BreastW	92.5±2.1	**93.9±0.5**
Ionosphere	88.1±1.3	**91.8±1.1**
Pima	40.5±4.7	**55.5±1.2**
Satellite	61.1±2.2	**71.1±1.7**
Average	70.6	**78.1**

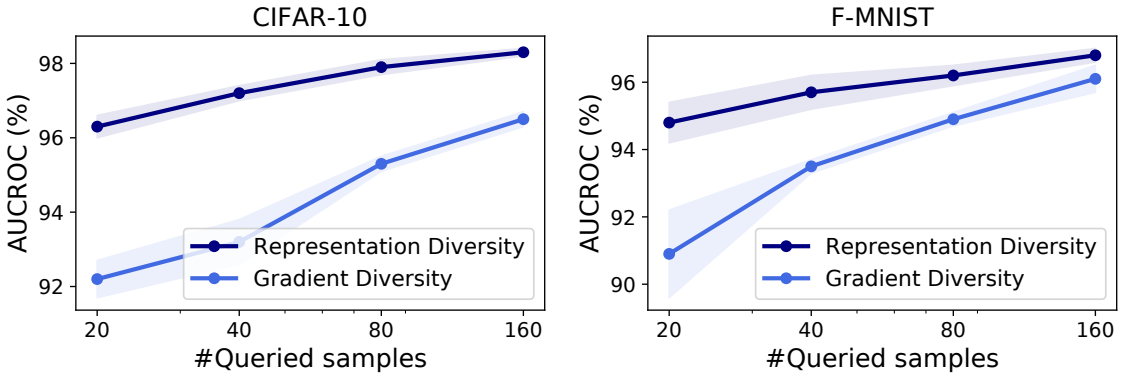


Figure C.11: Comparison with gradient diversity querying strategy (BADGE) [Ash et al., 2020]. The gradients wrt. the penultimate layer representation don't provide as informative queries as the representation itself, thus outperformed by our querying strategy SOEL. The true contamination ratio is 10%.

Comparisons to kNN [Ramaswamy et al., 2000] We compared against kNN in two ways. First we confirmed that our baseline backbone model NTL is competitive with kNN, which is shown to have a strong performance on tabular data [Shenkar and Wolf, 2022]. To this end, NTL has been shown to yield 95.7% AUC on clean CIFAR-10 data, see Shenkar and Wolf, 2022, Table 1 column 1. In contrast, Qiu et al. [2022b] reported 96.2% AUC in Table 2, which is very close.

Second, we tested the performance of the kNN method on our corrupted training data set. We gave kNN the advantage of using the ground truth contamination ratio (otherwise when

under-estimating this value, we saw the method degrade severely in performance).

KNN has two key hyperparameters: the number of nearest neighbors k and the assumed contamination ratio of the training set. The method uses this assumed contamination ratio when fitting to define the threshold on the decision function. In our experiments, we tried multiple values of k and reported the best testing results. Although the ground truth anomaly rate is unknown and our proposed methods don't have access to it, we gave kNN the competitive advantage of knowing the ground truth contamination ratio.

We studied the same tabular data sets as in our paper: BreastW, Ionosphere, Pima, and Satellite. We used the same procedure for constructing contaminated data outlined in our paper, where the contamination ratio was set to 10%. The results are summarized in Table C.7.

We adopted PyOD's implementation of kNN[5] and set all the other hyperparameters to their default values ("method, radius, algorithm, leaf_size, metric, p, and metric_params"). We repeated the experiments 10 times and reported the mean and standard deviation of the F1 scores in Table C.7. We find that our active learning framework outperforms the kNN baseline.

In more detail, the F1 scores for different values of k are listed below, where $k = 1, 2, 5, 10, 15, 20$, respectively:

- BreastW: 84.3±7.6, 86.5±3.1, 89.9±3.9, 90.7±3.4, 92.5±2.1, 91.9±1.5
- Ionosphere: 88.1±1.3, 87.6±2.6, 84.5±3.9, 75.2±2.5, 70.4±3.6, 67.4±3.4
- Pima: 34.4±3.6, 32.3±3.4, 36.9±6.4, 40.5±4.7, 35.3±3.6, 35.5±4.5
- Satellite: 51.0±1.1, 53.5±0.7, 54.7±1.3, 57.4±1.8, 59.3±1.3, 61.1±2.2

[5] `https://github.com/yzhao062/pyod`

Comparisons to Gradient Diversity Querying Strategy (BADGE) [Ash et al., 2020] We compared against a popular active learning method, BADGE [Ash et al., 2020], which is a diversity-driven active learning method that exploits sample-wise gradient diversity. We start with observing that BADGE doesn't work well for anomaly detection in Figure C.11, where we only replaced the objects that k-means++ works on in SOEL with gradients demanded in BADGE [Ash et al., 2020] while keeping all other settings fixed. This variant is referred to as "Gradient Diversity" while ours is denoted by "Representation Diversity". Figure C.11 shows the performance of Gradient Diversity is outperformed by a large margin, failing in querying informative samples as our Representation Diversity.

To understand which part of BADGE breaks for anomaly detection tasks, we check the gradients used by BADGE in an anomaly detection model. Before that, we start with describing how BADGE works. BADGE is developed for active learning in classification tasks. Given a pre-trained classifier, it first predicts the most likely label $\hat{y}$ (pseudo labels) for the unlabeled training data $\mathbf{x}$. These pseudo labels are then used to formulate a cross entropy loss $l_{CE}(\mathbf{x}, \hat{y})$. BADGE computes every data point's loss function's gradient to the final layer's weights as the data's representation. Upon active querying, a subset of data are selected such that their representations are diverse. In particular, the gradient to each class-specific weight W_k is $\nabla_{W_k} l_{CE}(\mathbf{x}, \hat{y}) = (p_k - \mathbb{1}(\hat{y} = k))\phi(\mathbf{x})$ where p_k is the predicted probability of being class k and $\phi(\mathbf{x})$ is the output of the penultimate layer. Proposition 1 of Ash et al. [2020] shows the norm of the gradient with pseudo labels is a lower bound of the one with true labels. In addition, note that the gradient is a scaling of the penultimate layer output. The scaling factor describes the predictive uncertainty and is upper bounded by 1. Therefore, the gradients are informative surrogates of the penultimate layer output of the network, as shown by the inequality

$$||\nabla_{W_k} l_{CE}(\mathbf{x}, \hat{y})||^2 \leq ||\nabla_{W_k} l_{CE}(\mathbf{x}, y)||^2 \leq ||\phi(\mathbf{x})||^2. \tag{C.4}$$

However, these properties are associated with the softmax activation function usage. In anomaly detection, models and losses are diverse and are beyond the usage of softmax activation outputs. Hence the gradients are no longer good ways to construct active queries. For example, the supervised deep SVDD [Ruff et al., 2019] uses the contrasting loss $l(\mathbf{x}, y) = y/(W\phi(\mathbf{x}) - \boldsymbol{c})^2 + (1-y)(W\phi(\mathbf{x}) - \boldsymbol{c})^2$ to compact the normal sample representations around center $\boldsymbol{c}$. However, the gradient $\nabla_W l(\mathbf{x}, y) = \big(2(1-y)(W\phi(\mathbf{x}) - c) - 2y(W\phi(\mathbf{x}) - c)^{-3}\big)\phi(\mathbf{x})$ is not a bounded scaling of $\phi(\mathbf{x})$ any more, thus not an informative surrogate of point $\mathbf{x}$.

C.5.14 NTL as a Unified Backbone Model

In Section 4 of the main paper, we have empirically compared SOEL to active-learning strategies known from various existing papers, where these strategies originally were proposed using different backbone architectures (either shallow methods or simple neural architectures, such as autoencoders). However, several recent benchmarks have revealed that these backbones are no longer competitive with modern self-supervised ones [Alvarez et al., 2022]. For a fair empirical comparison of SOEL to modern baselines, we upgraded the previously proposed active-learning methods by replacing their simple respective backbones with a modern self-supervised backbone: NTL [Qiu et al., 2021]—the same backbone that is also used in SOEL.

We motivate our choice of NTL as unified backbone in our experiments as follows. Figure C.12 shows the results of ten shallow and deep anomaly detection methods [Tax and Duin, 2004a, Liu et al., 2008, Diederik P. Kingma, 2014, Makhzani and Frey, 2015, Deecke et al., 2018, Ruff et al., 2018, Golan and El-Yaniv, 2018, Hendrycks et al., 2019, Sohn et al., 2020b, Qiu et al., 2022b] on the CIFAR10 one-vs.-rest anomaly detection task. NTL performs best (by a large margin) among the compared methods, including many classic backbone models known from the active anomaly detection literature [Görnitz et al., 2013, Barnabé-

Lortie et al., 2015, Das et al., 2019, Ruff et al., 2019, Pimentel et al., 2020, Trittenbach et al., 2021, Ning et al., 2022].

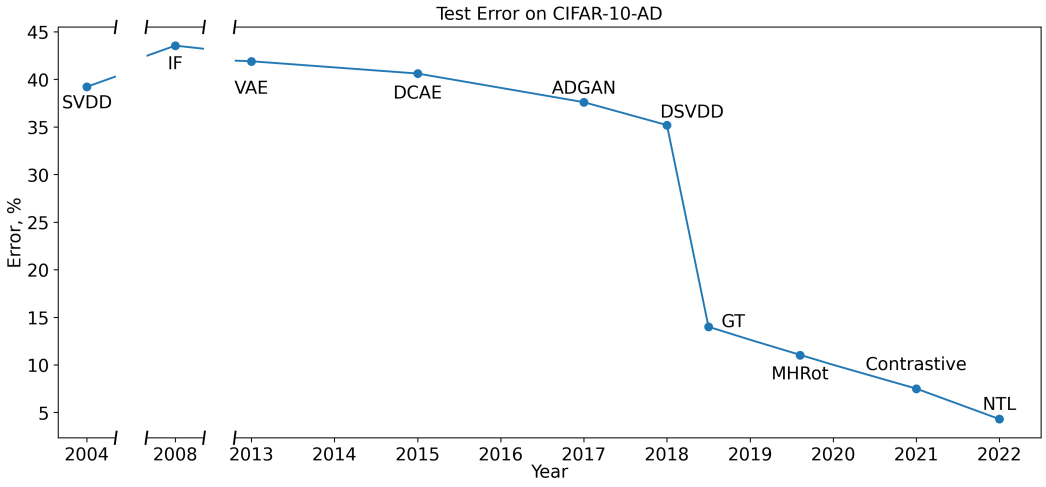


Figure C.12: Error (in % of 1-AUCROC) of ten methods on CIFAR10: two shallow methods (SVDD [Tax and Duin, 2004a] and IF [Liu et al., 2008]) and eight deep methods (VAE [Diederik P. Kingma, 2014], DCAE [Makhzani and Frey, 2015], ADGAN [Deecke et al., 2018], DSVDD [Ruff et al., 2018], GT [Golan and El-Yaniv, 2018], MHRot [Hendrycks et al., 2019], Contrastive [Sohn et al., 2020b], and NTL [Qiu et al., 2022b]). NTL achieves the best anomaly detection performance on CIFAR10.

An independent benchmark comparison of 13 methods (including nine deep methods proposed in 2018–2022) [Alvarez et al., 2022] recently identified NTL as the leading anomaly-detection method on tabular data. In their summary, the authors write: 'NeuTraLAD, the transformation-based approach, offers consistently above-average performance across all datasets. The data-augmentation strategy is particularly efficient on small-scale datasets where samples are scarce.'. Note that the latter is also the scenario where active learning is thought to be the most promising. We show the results from Alvarez et al. [2022] in Table C.8.

Table C.8: F1-scores (in %) and their standard deviations of 13 anomaly detection methods on tabular data. Results are taken from Alvarez et al. [2022]. The results indicate that NTL is the state-of-the-art for tabular anomaly detection.

	KDDCUP10	NSL-KDD	IDS2018	Arrhythmia	Thyroid	Avg.
ALAD	95.9±0.7	92.1±1.5	59.0±0.0	57.4±0.4	68.6±0.5	74.6
DAE	93.2±2.0	**96.1±0.1**	**71.5±0.5**	61.5±2.5	59.0±1.5	76.3
DAGMM	95.9±1.4	85.3±7.4	55.8±5.3	50.6±4.7	48.6±8.0	67.2
DeepSVDD	89.1±2.0	89.3±2.0	20.8±11	55.5±3.0	13.1±13	53.6
DROCC	91.1±0.0	90.4±0.0	45.6±0.0	35.8±2.6	62.1±10	65.0
DSEBM-e	96.6±0.1	94.6±0.1	43.9±0.8	59.9±1.0	23.8±0.7	63.8
DSEBM-r	**98.0±0.1**	95.5±0.1	40.7±0.1	60.1±1.0	23.6±0.4	63.6
DUAD	96.5±1.0	94.5±0.2	**71.8±2.7**	60.8±0.4	14.9±5.5	67.7
MemAE	95.0±1.7	95.6±0.0	59.9±0.1	62.6±1.6	56.1±0.9	73.8
SOM-DAGMM	97.7±0.3	95.6±0.3	44.1±1.1	51.9±5.9	52.7±12	68.4
LOF	95.1±0.0	91.1±0.0	63.8±0.0	61.5±0.0	68.6±0.0	76.0
OC-SVM	96.7±0.0	93.0±0.0	45.4±0.0	**63.5±0.0**	68.1±0.0	73.3
NTL	96.4±0.2	**96.0±0.1**	59.5±8.9	60.7±3.7	**73.4±0.6**	**77.2**

Appendix D

Chapter 5

D.1 Justifications of Assumptions A1-A3

As follows, we provide justifications for assumptions A1-A3. Following the justification, we also discuss possibilities to remove or mitigate the assumptions.

A1 Assuming an available meta-training set is widely adopted in few-shot learning or meta-learning [Finn et al., 2017, Nichol et al., 2018, Frikha et al., 2021, Huang et al., 2022] and domain generalization [Li et al., 2018a, Xiao et al., 2023]. In practice, the meta-training set can be generated using available covariates. For example, for our tabular data experiment, we used the timestamps; in medical data, one could use data collected from different hospitals or different patients to obtain separate sets for meta-training; and in MVTec-AD, we used the other training classes except for the target class to form the training set. We also provided an ablation study on the number of classes in the meta-training set (Table D.4). We found even in the extreme case where we only have one data class in the training set, the trained model still provides meaningful results.

There are multiple ways to mitigate this assumption. If one does not have a meta-training set at hand, one can train their model on a different but related dataset, e.g., train on Omniglot but test on MNIST (see results below under this setting. We still get decent AUC results on MNIST).

Anomaly ratio	1%	5%	10%	20%
AUROC	84.4±2.4	85.2±2.5	84.3±2.5	82.2±2.4

A2 Batch-level prediction is a common assumption used in robustness literature [Schneider et al., 2020, Nado et al., 2020, Lim et al., 2023, Wang et al., 2021, Choi et al., 2022]. In addition, batch-level predictions are widely used in real life. For example, people examine Covid19 test samples at a batch level out of economic and time-efficiency considerations[1]. To relax this assumption, our method can easily be extended to score individual data by pre-setting the sample mean and variance in BatchNorm layers with a collection of data. These moments are then fixed when predicting new individual data. Empirically, to understand the impacts of the batch size on the prediction performance, we conducted an ablation study with as small a batch size as three in the experiments.

A3 Besides being supported by the intuition that anomalies are rare, this is consistent with most of the data used in the literature. ADBench[2] has 57 anomaly detection datasets (with an average anomaly ratio of 5%), all matching our assumption that the normal data take the majority in each dataset.

We provide a simple mathematical argument for the validity of **A3**, showing that a mini-batch with a majority of anomalies is very unlikely to be drawn for a sufficiently large mini-batch size B. Let $p < 1/2$ denote the fraction of anomalies among the data and define

[1] `https://www.fda.gov/medical-devices/coronavirus-covid-19-and-medical-devices/pooled-sample-testing-and-screening-testing-covid-19`

[2] `https://github.com/Minqi824/ADBench`

$\Delta = 1/2 - p > 0$. For every data point $\mathbf{x}_i$ in the batch, let $y_i \sim \text{Bernoulli}(p)$ encode whether $\mathbf{x}_i$ is normal ($y_i = 0$) or abnormal ($y_i = 1$). The variable $S_B := y_1 + \cdots + y_B$ thus counts the number of anomalies in the batch, so that $S_B < B/2$ means that the majority of the data is normal. We want to show that the violation of A3 is unlikely, that is, $P(S_B \geq B/2)$ is small. By Hoeffding's inequality, since $0 \leq y_i \leq 1$ for all i, it follows that $P(S_B \geq B/2) = P(S_B - \mathbb{E}[S_B] \geq B/2 - Bp) \leq \exp\big(-2B(0.5-p)^2\big) \leq \exp\big(-2B\Delta^2\big)$, which converges to zero exponentially fast when $B \to \infty$.

D.2 Generalization to an Unseen Distribution P_*

This section aims to provide a proof for Theorem 5.1. Inspired by Fallah et al. [2021], we derive an upper bound of the generalization error of our meta-training approach on unseen distributions. The error is described in terms of the data distributions transformed by batch-norm-involved feature extractors.

Definition D.1. *Given a sample space Ω and its σ-field $\mathcal{F}$, the total variation distance between two probability measure P_i and P_j defined on $\mathcal{F}$ is*

$$\|P_i - P_j\|_{TV} = \sup_{A \in \mathcal{F}} |P_i(A) - P_j(A)| = \sup_{f: 0 \leq f \leq 1} \big|\mathbb{E}_{x \sim P_i}[f(x)] - \mathbb{E}_{x \sim P_j}[f(x)]\big| \tag{D.1}$$

Now we split the loss function into two parts: the first part is the layers before (including) the last batch normalization layer, referred to as feature extractor $\mathbf{z} = f_\theta(\mathbf{x})$, and the second part is the layers after the last batch normalization layer, namely the loss function map $\mathbf{L}(\mathbf{z}) = \mathbf{L}(f_\theta(\mathbf{x})) = \mathbf{L}^\theta(\mathbf{x})$. $\mathbf{L}(\mathbf{z})$ may involve learnable parameters, but we omit the notations for conciseness. When the input is only an individual data point, and there are no batch effects, we write $\mathcal{L}(\mathbf{z})$ to differentiate the vector-valued loss. The split allows us to separate the effects of batch normalization layers on the generalization error of unseen distributions.

Under the transformation of f_θ consisting of batch normalization layers, we have the data distribution transformed into the distribution of adaptatively centered representations

$$P_j(\mathbf{x}) \overset{\mathbf{z}=f_\theta(\mathbf{x})}{\Longrightarrow} P_j^z(\mathbf{z}), \quad j = 1, \cdots, K, * \tag{D.2}$$

resulting in P_j^z with $\mathbb{E}_{P_j^z}[\mathbf{z}] = 0$ and $\mathrm{Var}_{P_j^z}[\mathbf{z}] = 1$.

Assume the mini-batch is large enough so that the mini-batch means and variances are approximately constant across batches, i.e., the batch statistics in batch normalization layers are equal to the population-truth values. Consequently, when $\mathbf{x}_1, \ldots, \mathbf{x}_B \overset{\text{i.i.d.}}{\sim} P_j$ which constitutes $\mathbf{x}_\mathcal{B}$, their latent representations $\mathbf{z}_i := \mathbf{f}_\theta^i(\mathbf{x}_\mathcal{B}) \overset{\text{i.i.d.}}{\sim} P_j^z$. for $i = 1, \cdots, B$. Then the expectation of the batch-level losses are

$$\begin{aligned}
\mathbb{E}_{\mathbf{x}_\mathcal{B} \sim P_j}\left[\frac{1}{B}\sum_{i=1}^{B} \mathbf{L}_i^\theta(\mathbf{x}_\mathcal{B})\right] &= \mathbb{E}_{\{\mathbf{z}_i \sim P_j^z\}_{i=1}^B}\left[\frac{1}{B}\sum_{i=1}^{B} \mathcal{L}(\mathbf{z}_i)\right] \\
&= \frac{1}{B}\sum_{i=1}^{B} \mathbb{E}_{\{\mathbf{z}_i \sim P_j^z\}_{i=1}^B}\left[\mathcal{L}(\mathbf{z}_i)\right] \\
&= \mathbb{E}_{\mathbf{z} \sim P_j^z}\left[\mathcal{L}(\mathbf{z})\right]
\end{aligned} \tag{D.3}$$

Assumption D.1. For any parameters (if any), the loss function $\mathcal{L}$ is bounded by C.

We now quantify the generalization error to an unseen distribution P_* by the difference

between the expected loss of data batches of P_* and the one of meta-training distribution.

$$\left|\mathbb{E}_{\mathbf{x}_\mathcal{B}\sim P_*}\left[\frac{1}{B}\sum_{i=1}^{B}\mathbf{L}_i^\theta(\mathbf{x}_\mathcal{B})\right]-\frac{1}{K}\sum_{j=1}^{K}\mathbb{E}_{\mathbf{x}_\mathcal{B}\sim P_j}\left[\frac{1}{B}\sum_{i=1}^{B}\mathbf{L}_i^\theta(\mathbf{x}_\mathcal{B})\right]\right| \tag{D.4}$$

$$=\left|\mathbb{E}_{\mathbf{z}\sim P_*^z}\left[\mathcal{L}(\mathbf{z})\right]-\frac{1}{K}\sum_{j=1}^{K}\mathbb{E}_{\mathbf{z}\sim P_j^z}\left[\mathcal{L}(\mathbf{z})\right]\right| \quad \text{(by Equation (D.3))} \tag{D.5}$$

$$=C\left|\mathbb{E}_{\mathbf{z}\sim P_*^z}\left[\frac{\mathcal{L}(\mathbf{z})}{C}\right]-\frac{1}{K}\sum_{j=1}^{K}\mathbb{E}_{\mathbf{z}\sim P_j^z}\left[\frac{\mathcal{L}(\mathbf{z})}{C}\right]\right| \quad \text{(by Assumption D.1)} \tag{D.6}$$

$$\leq C\sup_{0\leq\mathcal{L}/C\leq 1}\left|\mathbb{E}_{\mathbf{z}\sim P_*^z}\left[\frac{\mathcal{L}(\mathbf{z})}{C}\right]-\frac{1}{K}\sum_{j=1}^{K}\mathbb{E}_{\mathbf{z}\sim P_j^z}\left[\frac{\mathcal{L}(\mathbf{z})}{C}\right]\right| \tag{D.7}$$

$$=C\left\|P_*^z-\frac{1}{K}\sum_{j=1}^{K}P_j^z\right\|_{TV} \quad \text{(by Definition D.1)} \tag{D.8}$$

This result suggests the generalization error of the loss function is bounded by the total variation distance between P_*^z and $\frac{1}{K}\sum_{j=1}^{K}P_j^z$. The batch normalization re-calibrates *all* P_j such that P_j^z centers at the origin and has unit variance, making the distributions similar. Thus the total variation gets smaller after batch normalization, lowering the generalization error upper bound.

The limitation of this analysis is we assume the batch statistics are population-truth moments (mean and variance) in $\mathbf{z}_i \overset{\text{i.i.d.}}{\sim} P_j^z$. So we cannot analyze the effects of the batch size B during training and testing. That said, we provide empirical evaluations on different batch sizes B at test time in Appendix C.5.

D.3 Algorithm

The training procedure of our approach is simple and similar to any stochastic gradient-based optimization. The only modification is to take into account the existence of a meta-training set. See Algorithm 3.

Algorithm 3: Training procedure of ACR

Input : K interrelated training distributions $P_1, \cdots, P_K \overset{\text{i.i.d.}}{\sim} \mathcal{Q}$
Mixting rate π
Deep anomaly detector model parameters θ
Sub-sample size M
Mini-batch size $|\mathcal{B}|$
learning rate α
Number of training iterations T

Output: Optimized anomaly detector with parameter θ_T

Randomly initialize θ
Construct $P_1^\pi, \cdots, P_K^\pi$ (Equation (5.6))
for *iteration* t *in* $[1, \cdots, T]$ **do**
  Sample M tasks $\{\mathbf{x}_{\mathcal{B}_m}\}_{m=1}^M$ from all K task distributions $\{P_1^\pi, \cdots, P_K^\pi\}$
  $\theta_t \leftarrow \theta_{t-1} - \alpha \nabla_\theta \frac{1}{M} \sum_{m=1}^M \mathbf{L}^{\theta_{t-1}}(\mathbf{x}_{\mathcal{B}_m}))$ (Equation (5.7))
end

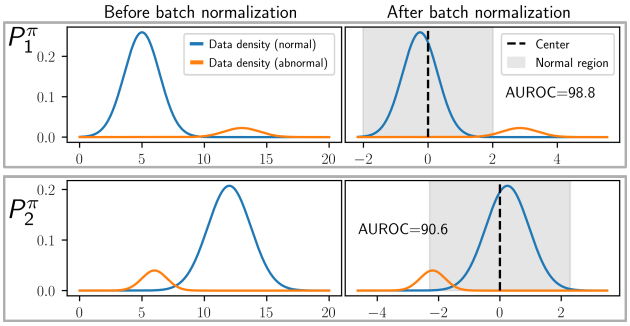


Figure D.1: Illustration of batch normalization for AD with two tasks P_1^π and P_2^π. The method (batch-)normalizes the data in P_j^π separately. If each P_j^π consists mainly of normal samples, most samples will be shifted close to the origin (by subtracting the respective task's mean). As a result, the samples from all tasks concentrate around the origin in a joint feature space (gray area) and thus can be tightly enclosed using, e.g., one-class classification. Samples from the test task are batch normalized in the same way.

D.4 Toy Example with Batch Normalization

An important component of our method is batch normalization, which shifts and re-scales any data batch $\mathbf{x}_\mathcal{B}$ to have sample mean zero and variance one. Batch normalization also provides a basic parameter-free zero-shot batch-level anomaly detector (Equation (5.4)). In Figure D.1, we show a 1D case of detecting anomalies in a mixture distribution. The mixture distribution composes of a normal data distribution (the major component) and an abnormal

data distribution (the minor component). Equation (5.4) adaptively detects anomalies at a batch level by shifting the normal data distribution toward the origin and pushing anomalies away. Setting a user-specified threshold allows making predictions.

D.5 Baselines

CLIP-AD [Liznerski et al., 2022]. CLIP (Contrastive Language–Image Pre-training [Radford et al., 2021]) is a pre-trained visual representation learning model that builds on open-source images and their natural language supervision signal. The resulting network projects visual images and language descriptions into the same feature space. The pre-trained model can provide meaningful representations for downstream tasks such as image classification and anomaly detection. When applying CLIP on zero-shot anomaly detection, CLIP prepares a pair of natural language descriptions for normal and abnormal data: $\{l_n = \text{“A photo of \{NORMAL\_CLASS\}”}, l_a = \text{“A photo of something”}\}$. The anomaly score of a test image $\mathbf{x}$ is the relative distance between $\mathbf{x}$ to l_n and $\mathbf{x}$ to l_a in the feature space,

$$s(\mathbf{x}) = \frac{\exp(\langle f_x(\mathbf{x}), f_l(l_a)\rangle)}{\sum_{c\in\{l_n,l_a\}} \exp(\langle f_x(\mathbf{x}), f_l(c)\rangle)},$$

where f_x and f_l are the CLIP image and description feature extractors and $\langle\cdot,\cdot\rangle$ is the inner product. We name this baseline CLIP-AD.

Compared to our proposed method, CLIP-AD requires a meaningful language description for the image. However, this is not always feasible for all image datasets like Omniglot [Lake et al., 2015], where people can't name the written characters easily. In addition, CLIP-AD doesn't apply to other data types like tabular data or time-series data. Finally, CLIP-AD has limited ability to adapt to a different data distribution other than its training one. These

limitations are demonstrated in our experiments.

OC-MAML [Frikha et al., 2021]. One-Class Model Agnostic Meta Learning (OC-MAML) is a meta-learning algorithm that taylors MAML [Finn et al., 2017] toward few-shot anomaly detection setup. OC-MAML learns a global model parameterization θ that can quickly adapt to unseen tasks with a few new task data points S, called a support set. The new-task adaptation takes the global model parameters to a task-specific parameterization $\phi(\theta, S_t)$ that has a low loss $L(Q_t; \phi(\theta, S_t))$ on the new task t, represented by another dataset Q_t, called a query set. OC-MAML uses a one-class support set to update the model parameters θ with a few gradient steps to get ϕ. To learn an easy-to-adapt global parameterization θ, OC-MAML directly minimizes the target loss on lots of training tasks. Suppose there are T tasks for training. The following loss function is minimized

$$l(\theta) = \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{S_t \sim p_S^t, Q_t \sim p_Q^t}[L(Q_t; \phi(\theta, S_t))], \tag{D.9}$$

where p_S^t is task t's support set distribution and p_Q^t is the query set distribution. During training, the support set contains K normal data points where K is usually small, termed K-shot OC-MAML. The query set contains an equal number of normal and abnormal data and provides optimization gradients for θ. During test time, OC-MAML adapts the global parameter θ on the unseen task's support set S^*, resulting in a task-specific parameter $\phi(\theta, S^*)$. The newly adapted parameters are then used for downstream tasks.

OC-MAML is not a zero-shot anomaly detector and requires K support data points to adapt compared to our method. Our method is simpler in training as it doesn't need to adapt to the support set with additional gradient updates, characterized in the function $\phi(\theta, S)$. OC-MAML is also different in batch normalization. Rather than the original batch normalization, OC-MAML first computes the batch moments using the support set and then normalizes both the support and query set with the same moments. However, the computed

moments can be noisy when the support set size is small. In our experiments, we adopt a 1-shot OC-MAML for all image data.

ResNet152 [He et al., 2016]. Because batch normalization is an effective tool for zero-shot anomaly detection (see Figure 5.1b), we directly apply batch normalization on extracted features from a pre-trained model. We then compute the anomaly score as the Euclidean distance between a feature vector and the origin in the feature space. Our experiments use a ResNet152 model pre-trained on ImageNet as a feature extractor and extract its 2048-dimension penultimate layer output as the final feature vector. Upon computing the features of an input batch through batch normalization layers in ResNet, two variants are available: using the batch statistics from training or re-computing the statistics of the test input batch itself. We name the former variant ResNet152-I and the latter ResNet152-II. Baseline ResNet152 doesn't optimize the feature extractor jointly with the zero-shot detection property of batch normalization. Hence the extracted pre-trained features are not optimal for the zero-shot anomaly detection.

ADIB [Deecke et al., 2021]. In addition to zero-shot and few-shot anomaly detectors, we also compare with the state-of-the-art deep anomaly detector ADIB [Deecke et al., 2021] which use pre-trained image features and additional data for outlier exposure in training. We use a "debiased" subset of TinyImageNet as the outlier exposure data for CIFAR100 as suggested in Hendrycks et al. [2018], use EMNIST [Cohen et al., 2017] as the outlier exposure data for MNIST as suggested in Liznerski et al. [2022], use OrganC and OrganS datasets [Yang et al., 2021a] as outlier exposure data for OrganA, and use half of the training data as normal data and half of the training data as auxiliary outliers for Omniglot.

D.6 Implementation Details

Practical Training and Testing. On visual anomaly classification and tabular anomaly detection, we construct training and test distributions using labeled datasets[3], where all $\mathbf{x}$ from the same class j (e.g., all 0's in MNIST) are considered samples from the same P_j. The dataset $\mathcal{Q}$ (e.g., MNIST as a whole) is the meta-set of all these distributions.

For training and testing, we split the meta-dataset into disjoint subsets. In the MNIST example, we define $P_0, ..., P_4$ as the distributions of images with digits $0-4$ and use them for training. For testing, we select a single distribution of digits not seen during training (e.g., digit 5) as the "new normal" distribution P_* to which we adapt the model. The remaining digits ($6-9$ in this example) are used as test-time anomalies. To reduce variance, we rotate the roles among digits $5-9$, using each digit as a test distribution once.[4]

D.6.1 Implementation Details on Image Data for Anomaly Detection

Hyperparameter Search. We search the hyperparameters on a validation set split from the training set, after which we integrate the validation set into the training set and train the model on that. Then we test the model on the test set.

On CIFAR100-C, we construct the validation set on the training set of the primitive CIFAR100. We randomly select 20 classes as the validation set and set the remaining classes to be the training dataset at validation time. We search the neural network architecture (layers (3,4,5,6), number of convolutional kernels (32, 64, 128), and the output dimensions (8, 16, 32, 64, 128) while fixing the kernel size by 3x3. For the learning rate, we search values

[3] these are either classification datasets or datasets where one of the covariates is binned to provide classes.

[4] This is the popular "one-vs-rest" testing set-up, which is standard in anomaly detection benchmarking. (e.g., [Ruff et al., 2021])

0.1, 0.01, 0.001, 0.0001, and 0.00001, after which we search finer values in a binary search fashion. We also search the mini-batch size B (30, 60) and the number of sub-sampled tasks M (16, 32, 64) at each iteration. We select the combination that trade-off the convergence speed and optimization stability. When selecting the anomaly ratio π (Equation (5.6)), we test 0.99, 0.95, 0.9, 0.8, 0.6 and find the results are quite robust to these values. So we fix $\pi = 0.8$ across the experiments.

On non-natural image datasets (OrganA, Omniglot, and MNIST), we search hyperparameters on the validation set of Omniglot and use the searched hyperparameters on all datasets. Specifically, at validation time, we randomly split the Omniglot into 1200 classes for training and 423 classes for validation. After optimizing the hyperparameters, we constantly use the first 1200 classes for training and the remaining 423 classes for testing. The searched hyperparameters are the same as the ones in CIFAR100-C described above.

Training Protocols. We train the model 6,000 iterations on CIFAR100 data, 10,000 iterations on Omiglot, and 2,000 iterations on MNIST and OrganA. Each iteration contains 32 training tasks; each task mini-batch has 30 (for datasets other than CIFAR100) or 60 (for CIFAR100) points sampled from $P_j^{0.8}$. All 32 training tasks' gradients are averaged and incur one gradient update per iteration.

ACR-DeepSVDD. We use the standard convolutional neural network architecture used in meta-learning. Specifically, the network contains four convolution layers. Each convolution layer is followed by a batch normalization layer and a ReLU activation layer. The final layer is a fully-connectly layer followed by a batch normalization layer. The center of DSVDD has the same dimension as the output of the fully-connected layer, which is 32. For CIFAR100/CIFAR100-C, each convolution layer has 128 kernels. For MNIST, Omniglot, and OrganA, each convolution layer has 64 kernels. Each kernel's size is 3x3. We use Adam

with a learning rate of 0.003 on CIFAR100 dataset and $1e-4$ on all the other datasets.

ACR-BCE. We use the same network structure as ACR-DeepSVDD without the final batch normalization layer and the center. The final fully-connected layer has output dimension of 1. We train the model with binary cross entropy loss. We use Adam with a learning rate of 0.003 on CIFAR100 dataset and $1e-4$ on all the other datasets.

D.6.2 Implementation Details on MVTec AD for Anomaly Segmentation

Training Protocols. Since the images are roughly aligned, we can use a sliding window over the image to detect local defects in each window. However, this requires pixel-level alignment and is unrealistic for the MVTec AD dataset. We instead first extract informative texture features using a sliding window, which corresponds to the 2D convolutions. The convolution kernel is instantiated with the ones in a pre-trained ResNet. We follow the same data pre-processing steps of Cohen and Hoshen [2020], Rippel et al. [2021], Defard et al. [2021] to extract the texture representations (the third layer's output in our case) of WideResNet-50-2 pre-trained on ImageNet. Second, we detect anomalies in the extracted features in each sliding window position with our ACR method. Specifically, each window position corresponds to one image patch. We stack into a batch the patches taken from a set of images that all share the same spatial position in the image. For example, we may stack the top-left patch of all testing `wood` images into a batch and use ACR to detect anomalies in that batch. Finally, the window-wise anomaly scores are bilinearly interpolated to the size of the original image, i.e., the pixel-level anomaly scores.

Following the same data pre-processing steps of Cohen and Hoshen [2020], Rippel et al. [2021], Defard et al. [2021], we extract the texture representations (the third layer's output

in our case) of WideResNet-50-2 pre-trained on ImageNet. After feature extraction, a batch of B images leads to a representation of size $(B, C, H, W) := (B, 1024, 14, 14)$. These representations contain both textual and spatial information. During meta-training, we treat each spatial position as one new class so that there are $14 \times 14 = 196$ new classes for each original class (e.g., `wood`), and each new class contains B data points, each a 1024 long vector. As a result, the model (DSVDD in our usage) takes a batch of vectors of size $(B, 1024)$ as input and assigns anomaly scores to each vector within the batch. When adding synthetic abnormal data (Equation (5.6)), we use Gaussian noise corrupted input vectors rather than new class data, incorporating the fact that local defects result in similar feature vectors instead of globally different textures. Specifically, we add Gaussian noise sampled from $\mathcal{N}(0, 0.01I)$ and set $\pi = 0.5$ in Equation (5.6). Since there is no During testing, we batch each position (h, w) of all images and detect anomalies at position (h, w). Because the defects are local, and there is a possibility that there is no defect at some position (h_0, w_0), we manually add synthetic noisy vectors into the tested batch as the training procedure to ensure the images have a low anomaly score at (h_0, w_0). After getting anomaly scores, we remove the synthetic vector results, leading to scores of size $(B, 14, 14)$, and then upscale the scores into the original image size by bilinear interpolation. We acknowledge that using CutPaste [Li et al., 2021b] to generate more realistic synthetic abnormal samples is another option. We leave the investigation to future work.

ACR-DeepSVDD and Hyperparameter Search. Our model is a five-layer MLP with intermediate batch normalization layers and ReLU activations. The hidden sizes of the perceptrons are $[512, 256, 128, 64, 32]$. The center is size 32. The statistics of all batch normalization layers are computed on fly on the training/test batches. We average the gradients of 32 randomly sampled tasks for each parameter update. Each task contains 30 normal feature vectors and 30 noise-corrupted feature vectors. We train the model with Meta Outlier Exposure loss. We set the learning rate 0.0003 and iterate 50 updates for each

class. We search the hyperparameters on a test subset of `bottle` class (half of the original test set) and apply the same hyperparameters to all classes afterward.

D.6.3 Implementation Details on Tabular Data

ACR-NTL has the same model architecture as the baseline NTL, and ACR-DeepSVDD adds one additional batch normalization layer on top of the DeepSVDD baseline. Our algorithm is applicable to the existing backbone models without complex modifications.

ACR-DeepSVDD. ACR is applied to the backbone model DeepSVDD [Ruff et al., 2018]. The neural network of DeepSVDD is a four-layer MLP with intermediate batch normalization layers and ReLU activations. The hidden sizes on Anoshift dataset are $[128, 128, 128, 32]$. The hidden sizes on Malware dataset are $[64, 64, 64, 32]$. One batch normalization layer is added on the top of the network on Anoshift experiment. The statistics of all batch normalization layers are computed on fly on the training/test batches. We use Adam with a learning rate of $4e-4$ on Anoshift dataset and $1e-4$ on Malware dataset.

ACR-NTL. ACR is applied to the backbone model NTL [Qiu et al., 2021]. The shared encoder of NTL is a four-layer MLP with intermediate batch normalization layers and ReLU activations. The hidden sizes of the encoder are $[128, 128, 128, 32]$. The statistics of all batch normalization layers are computed on fly on the training/test batches. We set the number of neural transformations as 19. Each neural transformation is parametrized by a three-layer MLP of hidden size of 128 with ReLU activations. All networks are optimized jointly with Adam with a learning rate of $4e-4$.

D.7 Meta Outlier Exposure Avoids Trivial Solutions.

The benefit of the outlier exposure loss in meta-training is that the learning algorithm cannot simply learn a model on the *average* data distribution, i.e., without learning to adapt. This failure to adapt is a common problem in meta-learning. Our solution relies on using each training sample $\mathbf{x}_i$ in different contexts: depending on the sign of $y_{i,j}$, data point $\mathbf{x}_i$ is considered normal (when drawn from P_j) or anomalous (when drawn from $\bar{P}_j$). This ambiguity prevents the model from learning an average model over the meta data set and forces it to adapt to individual distributions instead.

For example, DeepSVDD with its original loss function may suffer from a trivial solution that maps any input data to the origin in the feature space and achieves the optimal zero loss [Ruff et al., 2018]. This trivial solution is also possible in our proposed meta-training procedure. But Meta Outlier Exposure gets rid of this trivial solution because mapping everything to the origin incurs an infinite loss on $\mathbf{A}_\theta$. Similar reasoning also applies to binary cross entropy loss.

D.8 Connections to Other Areas

Our problem setup and assumptions share similarities with other research areas but differences are also pronounced.

Connection to Batch Normalization-Based Test-time Adaptation (TTA). Many works for TTA feature batch-level predictions [Schneider et al., 2020, Nado et al., 2020, Lim et al., 2023, Wang et al., 2021, Choi et al., 2022] assumes its test-time data are corrupted but from the *same semantic classes* as the training data, but the zero-shot AD's test data can be drawn from a completely new class.

Connection to Unsupervised Domain Adaptation (UDA). Although our approach uses unlabeled data like the UDA setting [Kouw and Loog, 2019], UDA assumes the unlabeled data from the shifted domain is available during training and can be used to update the model parameters. But our method doesn't rely on the availability of novel data during training time and doesn't require updating the model parameters during test time.

Connection to Zero-shot Classification. Xian et al. [2018] explains the nature of zero-shot classification and writes that "the crux of the matter for all zero-shot learning methods is to associate observed and non-observed classes through some form of auxiliary information which encodes visually distinguishing properties of objects." The *auxiliary information* demands extra human annotations like picture attributes. In contrast, our method assumes a batch of test data without human annotations. The test distribution information is automatically contained in the batch statistics.

Connection to Meta-learning. Although we assume that there is a meta-training dataset available like meta-learning, we don't require a support set for updating the model during both training time and test time. The presence of a support set differentiates our method from meta-learning in many aspects. First, for the most well-known technique (MAML) in meta-learning, training requires second-order derivative information of the support set loss function, which is computationally expensive and slows the optimization. Second, it is unclear how to select the support set size for adaptation. Sometimes, it may require a large support set to achieve good adaptations. For example, OC-MAML needs at least a 10-shot support set to perform on par with our method on CIFAR100-C; Third, the support set requires labeled data. This already adds burdens to practitioners. Fourth, the model parameter updates require additional maintenance and extra cost during testing.

Connection to Contextual anomaly detection. Contextual AD considers a changing notion of normality based on context [Gupta et al., 2013, Shulman, 2019]. In contextual AD, the training and testing data are from the *same* data generating process, which involves

Table D.1: AUC (%) with standard deviation for anomaly detection on CIFAR100-C and Omniglot. As an ablation, rather than utilizing outlier exposure, we trained Zero-shot BN only on normal data of each task.

	CIFAR100-C (Gaussian Noise)				Omniglot		
	1%	5%	10%	20%	5%	10%	20%
One-class loss	72.2±2.2	73.9±1.4	74.2±0.9	73.8±0.3	96.2±1.0	96.4±0.8	96.2±0.8
(data-adapted) ResNet152	70.9±2.2	67.6±0.2	67.0±0.7	64.9±0.5	99.2±0.2	99.1±0.1	99.0±0.1

(hidden or observed) contextual variables controlling the generation. This is different from our setup. We tackle the problem when the training and testing data are from different data generating processes.

D.9 Additional Results

D.9.1 Ablation Study

Training with Different Losses. We study the benefits of using meta outlier exposure in Equation (5.5) and compare to a) using one-class classification loss $\mathcal{L}[\mathbf{S}_\theta(\mathbf{x}_\mathcal{B})] = \frac{1}{B}\sum_{i\in\mathcal{B}}\mathbf{S}^i_\theta(\mathbf{x}_\mathcal{B})$ with $\mathbf{S}^i_\theta(\mathbf{x}_\mathcal{B}) = ||\phi^i_\theta(\mathbf{x}_\mathcal{B}) - \mathbf{c}||^2$ where ϕ_θ is the feature map, b) (data-adapted) ResNet152. The data-adapted ResNet152 first learns the features by performing a multi-class classification task with the meta-training set. Then a batch normalization layer is applied on the top of penultimate layer representations for zero-shot anomaly detection. We train a 100-class classifier for CIFAR100C and Omniglot separately. Note that for Omniglot, we randomly sub-sample 100 classes from its 1400 training classes and train the classifier. From the results in Table D.1 we can see that both ablations perform competitive with ACR on the simple Omniglot dataset, but perform much worse compared to ACR on the complex CIFAR100-C dataset. In conclusion, using meta outlier exposure in training is favorable.

Table D.2: The effects of batch normalization for zero-shot anomaly detection. The first two columns show different combinations of batch normalization usage during training and testing. The third column answers the question whether the type of batchnorm usage works for zero-shot anomaly detection.

BatchNorm (train)	BatchNorm (test)	Work?
✓	✓	Yes
✓	✗	No
✗	✓	No
✗	✗	No

Table D.3: The effects of test time batch size on the results of zero-shot anomaly detection. We report the test results in AUC when the contamination ratio is set to 5% and 10%. The studies are conducted on the Gaussian noise version of CIFAR-100C. On the extreme batch sizes, each batch contains one anomaly.

Batch size	3	6	11	16
One anomaly	66.4±2.3	77.9±2.8	82.3±2.7	84.8±2.0

Batch size	20	40	60	80	100
5%	83.7±1.9	85.3±1.1	85.6±1.2	85.9±0.8	85.6±0.7
10%	84.5±1.5	86.1±0.8	85.7±0.7	85.8±0.5	85.8±0.6

Table D.4: The effects of the number of classes used in training on zero-shot anomaly detection. We report the test results in AUC when the contamination ratio is fixed to 10%. The studies are conducted on the Omniglot dataset.

#Training classes	1	2	5	10	15
AUROC	59.0±0.6	71.8±0.6	72.5±0.3	72.2±1.0	75.3±0.4

#Training classes	20	40	80	160	320	640	1200
AUROC	79.0±1.0	90.5±0.5	95.3±0.2	97.6±0.2	98.1±0.2	98.4±0.1	99.1±0.2

Training or Testing Without BatchNorm. We investigate whether training or testing without batch normalization works for zero-shot anomaly detection or not. To this end, we employ four different combinations of batch normalization usage during training and testing and check which combination works and which doesn't. We trained the models with the same meta-training procedure as what we used in Section 5.4.1 and tested on CIFAR100-C and Omniglot. We present the results in Table D.2. In the third column, "Yes" indicates the AUROC metric is significantly larger than 0.5, and therefore learns a

meaningful zero-shot anomaly detection model; "No" indicates the AUROC performance is around 0.5, which means the predicted anomaly scores are just random guesses and the model cannot be used for zero-shot anomaly detection. Table D.2 shows that only when the batch normalization is used both in training and testing, the zero-shot anomaly detection works. Otherwise, the meta-training procedure couldn't result in meaningful zero-shot anomaly detection representations.

Moreover, for the DeepSVDD model, we can theoretically show that training without batch normalization will not work with meta outlier exposure: the optimal loss function has nothing to do with zero-shot anomaly detection. Rather, the optimal loss is only related to the mixture weight π during training. Without loss of generality, suppose we have two training distributions P_1, P_2. We learn a Deep SVDD model parameterized by θ and c by the meta outlier exposure method,

$$
\begin{aligned}
& l(\theta, c) \\
&= \mathbb{E}_{x\sim P_1^\pi}\left[(1-y_1)(f_\theta(x)-c)^2 + \frac{y_1}{(f_\theta(x)-c)^2}\right] + \mathbb{E}_{x\sim P_2^\pi}\left[(1-y_2)(f_\theta(x)-c)^2 + \frac{y_2}{(f_\theta(x)-c)^2}\right] \\
&= \mathbb{E}_{x_1\sim P_1, x_2\sim P_2}\left[\pi(f_\theta(x_1)-c)^2 + \frac{1-\pi}{(f_\theta(x_2)-c)^2} + \pi(f_\theta(x_2)-c)^2 + \frac{1-\pi}{(f_\theta(x_1)-c)^2}\right] \\
&= \sum_{i=1}^{2} \mathbb{E}_{x_i\sim P_i}\left[\pi(f_\theta(x_i)-c)^2 + \frac{1-\pi}{(f_\theta(x_i)-c)^2}\right] \\
&\geq 4\sqrt{\pi(1-\pi)}
\end{aligned}
\tag{D.10}
$$

where $0.5 < \pi < 1$ implies the majority assumption and the equality holds when the model parameters (θ and c) are tuned such that $(f_\theta(x_i)-c)^2 = \sqrt{(1-\pi)/\pi}$ for any x_i. All data points will be put at the hypersphere's surface centered around c with a radius $\sqrt{(1-\pi)/\pi}$ in the feature space when the model is trivially optimized. However, the optimal loss has nothing to do with distinguishing different distribution's input data x in the feature space,

which is unlikely to produce useful representations for zero-shot anomaly detection.

On the other hand, if we apply batch normalization in the model f_θ, we will not have the optimal loss function irrelevant to distributions. To see this, note that batch normalization will shift the input toward the origin. Thus x_1 in training task P_1^π and x_2 in training task P_2^π should have similar representations as they both take the majority in each task. Similarly, x_2 in task P_1^π and x_1 in task P_2^π are minorities, thus mapped far away from the origin in the feature space. Therefore, the symmetry breaks in Equation (D.10), and the above trivial optimal loss disappears.

Ablation Study on Batch Sizes. To test the batch size effects, we add an ablation study on the Gaussian noise version of CIFAR-100C where we fix the anomaly ratio as 5% or 10% and try different batch sizes. The results are summarized in Table D.3. It shows that larger batch sizes lead to more stable results. The performance is similar when the batch size is larger than or equal to 40. Even with a batch size being 20, our results are still better than the best-performing baseline.

We also test extreme batch sizes being 3, 6, 11, and 16 where each batch contains one anomaly.

Ablation Study on Number of Training Classes. To analyze the effect of the number of training distributions on the zero-shot AD performance, we conducted experiments on Omniglot where we varied the number of available meta-training classes from 1, 2, 5, 10, 15, 20, 40, 80 to 640, 1200. We separately trained ACR-DSVDD on each setup and tested the resulting models on the test set that has a 10% ground-truth anomaly ratio. We repeated 5 runs of the experiment with random initialization and reported the AUROC results in Table D.4. It shows that using 320 available classes for this dataset is sufficient to achieve a decent zero-shot AD performance. The results also demonstrate that even though we have

only one class in the meta-training set, thanks to the batch norm adaptation, we can still get better-than-random zero-shot AD performance.

Ablation Study on Other Normalization Techniques. As follows, we report new experiments involving LayerNorm, InstanceNorm, and GroupNorm for zero-shot AD.

We stress that, while these methods may have overall benefits in terms of enhancing performance, they do not work in isolation in our zero-shot AD setup. A crucial difference between these methods and batch normalization is that they treat each observation individually, rather than computing normalization statistics across a batch of observations. However, sharing information across the batch (and this way implicitly learning about distribution-level information) was crucial for our method to work.

Our experiments (AUROC results in the table below) with DSVDD on the Omniglot dataset support this reasoning. Using these normalization layers in isolations yields to random outcomes (AUROC=50):

LayerNorm	InstanceNorm	GroupNorm
50.0±0.9	50.6±0.7	50.2±0.5

We also added a version of the experiment where we combined these methods with batch normalization in the final layer. The results dramatically improve in this case:

BatchNorm (BN)	LayerNorm + BN	InstanceNorm + BN	GroupNorm + BN
99.1±0.2	98.8±0.1	98.8±0.2	98.2±0.2

Experimental details: We use DSVDD as the anomaly detector and experiment on the Omniglot dataset. Each nonlinear layer of the feature extractor for DSVDD is followed by the respective normalization layer. We apply the same training protocol as Table D.6 in the

paper. For GroupNorm, we separate the channels into two groups wherever we apply group normalization.

Effects of BatchNorm (BN) Layer Position. We conducted additional experiments on two visual anomaly detection tasks – anomaly segmentation on the MVTec-AD dataset and object-level AD on CIFAR100C. We used the same DSVDD model architectures as used in Tables 1 and 2 as the backbone model, except that we switched off BN in all but one layer. For anomaly segmentation, there are five possible BN layer positions; and there are four positions for the object-level AD model. We switched off the BN layers in all but one position and then re-trained and tested the model with the same protocol used in our main paper (For CIFAR100C, we tested the model with the test data anomaly ratio of 10%). We iterate this procedure across all available BN layer positions. We repeat every experiment with different random seeds five times and report the mean AUROC and standard deviation. The results are summarized in the tables below, where a smaller value of the BN position corresponds to earlier layers (close to the input), and a larger value corresponds to later layers close to the output. The final column is copied from our results in Tables 1 and 2 where BN layers are on all available positions. For both MVTec-AD and CIFAR100C, we average the performance across all test classes.

Results on the two tasks have opposite trends regarding the effects of BN layer positions. Specifically, for anomaly segmentation on MVTec-AD, earlier BN layers are more effective, while for AD on CIFAR100C, later BN layers are more effective. This observation can be explained by the fact that anomaly segmentation is more sensitive to low-level features, while object-level AD is more sensitive to global feature representations. In addition, compared to the results in Tables 1 and 2 (copied to the last column in the table below), our results suggest that using BN layers at multiple positions does help re-calibrate the data batches of different distributions from low-level features (early layers) to high-level features (late layers)

and shows performance improvement over a single BN layer.

MVTec-AD						
BN Position	1	2	3	4	5	(1,2,3,4,5)
Pixel-level	80.8±1.9	69.6±1.4	73.9±0.9	63.6±1.6	60.9±0.8	92.5±0.2
Image-level	74.7±0.9	59.2±1.6	63.6±1.3	65.5±1.2	65.4±1.3	85.8±0.6

CIFAR100C					
BN Position	1	2	3	4	(1,2,3,4)
AUROC	61.4±0.5	61.0±0.9	68.2±0.9	68.9±1.1	85.9±0.4

Robustness of the mixing hyperparameter π in Equation (5.6). We conduct the following experiments with varying π. The experiment has the same setup as Table 1 on CIFAR100C with a testing anomaly ratio of 0.1. The results show that all tested π's results are over 84% AUC.

CIFAR100C					
π	0.99	0.95	0.9	0.8	0.6
AUROC	85.8±0.5	85.4±0.5	84.1±0.4	85.9±0.4	84.4±0.6

D.9.2 Visualization of ACR.

We provide a visualization of the learned representations from DeepSVDD on the Omniglot dataset as qualitative evidence in Figure D.2. We observe that even though the normal and abnormal data classes flip in two plots, the model learns to center the samples from the majority class and map the samples from the minority class away to the center in the embedding space. In conclusion, ACR is an easy-to-use zero-shot anomaly detection method

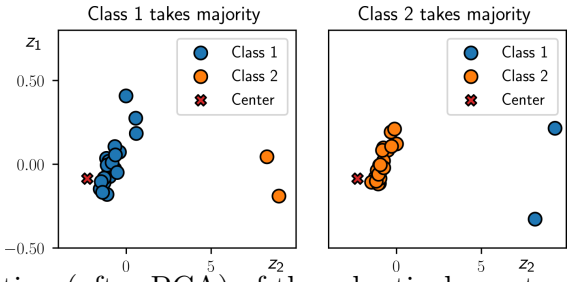


Figure D.2: 2D visualization (after PCA) of the adaptively centered representations for two test tasks in the Omniglot dataset. The same learned DeepSVDD model adapts with our proposed method and maps samples from the majority class (class 1 (left) and class 2 (right)) to the same center in the embedding space in both tasks.

and achieves superior zero-shot anomaly detection results on different types of images. The performance of ACR is also robust against the test anomaly ratios.

D.9.3 Additional Results on CIFAR100-C

We test all methods on all corruption types of CIFAR100-C. The results are presented in Table D.5.

D.9.4 Additional Results on Non-natural Images

Datasets. We further evaluate the methods on two other non-natural datasets–MNIST and Omniglot of hand-written characters. MNIST uses the same split and evaluation protocol as OrganA. On Omniglot, we take the first 1200 classes to form the meta-training set and use the remaining unseen 423 classes for testing.

Results. We present the results in Table D.6. It shows that our approach significantly outperforms all the other baselines by a large margin on both datasets.

D.9.5 Class-wise Results on MVTec-AD

We present the class-wise results in Table D.7 for finer comparisons with other methods on MVTec AD benchmark.

Besides, we also implement the synthetic anomalies using images from different distributions during training. The result is worse than the model using Gaussian corrupt noise as example anomalies but still better than existing works in anomaly segmentation. We summarize the results in Table D.8, which suggests that using images from different distributions as example anomalies during training is helpful for anomaly segmentation on MVTec-AD.

D.9.6 Additional Results on Malware

Dataset. Malware [Huynh et al., 2017] is a dataset of malicious and benign computer programs, collected from 11/2010 to 07/2014. Malware attacks are designed adversarially, thus leading to shifts in both normal and abnormal data. We adopt the data reader from Li et al. [2021a]. We follow the preprocessing of [Huynh et al., 2017] and convert the real-valued probabilities p of being malware to binary labels (labeled one if $p > 0.6$ and zero if $p < 0.4$). The samples with probabilities between 0.4 and 0.6 are discarded. The model is trained on normal samples collected from 01/2011 to 12/2013, validated on normal and abnormal samples from 11/2010 to 12/2010, and tested on normal and abnormal samples from 01/2014 to 07/2014 (the anomaly ratios vary between 1% and 20%).

Results. We report the results on Malware in Table D.9. ACR-NTL achieves the best results under all anomaly ratios. All baselines except ICL perform worse than random guessing, meaning that the malware successfully fools most baselines, which testifies to the adversarial-upgrade explanation in the main paper.

Table D.5: AUC (%) with standard deviation for anomaly detection on CIFAR100-C [Hendrycks and Dietterich, 2019].

Noise Type	Method	1%	5%	10%	20%
gaussian noise	ACR-DSVDD	87.7±1.4	86.3±0.9	85.9±0.4	85.6±0.4
	ACR-BCE	84.3±2.2	86.0±0.3	86.0±0.2	85.7±0.4
	ResNet152-I	75.6±2.3	73.2±1.3	73.2±0.8	69.9±0.6
	ResNet152-II	62.5±3.1	61.8±1.7	61.2±0.6	60.2±0.4
	OC-MAML (1-shot)	53.0±3.6	54.1±1.9	55.8±0.6	57.1±1.0
	CLIP-AD	82.3±1.1	82.6±0.9	82.3±0.9	82.6±0.1
shot noise	ACR-DSVDD	85.5±1.6	86.5±0.2	87.3±0.6	86.4±0.4
	ACR-BCE	87.1±2.4	86.3±0.6	86.8±0.5	86.4±0.1
	ResNet152-I	76.9±2.3	75.7±0.7	74.3±0.6	71.9±0.6
	ResNet152-II	59.7±2.0	60.9±1.4	61.0±0.6	60.1±0.6
	OC-MAML (1-shot)	53.8±4.7	52.8±1.1	53.6±1.0	53.8±1.3
	CLIP-AD	83.0±1.6	84.1±0.3	83.9±0.5	83.3±0.3
impulse noise	ACR-DSVDD	80.5±3.7	81.5±0.5	80.7±0.7	79.8±0.2
	ACR-BCE	81.7±1.0	81.0±0.5	80.8±0.7	79.5±0.3
	ResNet152-I	74.3±1.4	73.1±1.0	72.2±0.4	69.4±0.3
	ResNet152-II	64.3±2.7	63.0±1.2	62.2±0.8	61.2±0.6
	OC-MAML (1-shot)	53.6±2.5	54.8±1.6	53.6±1.1	53.8±0.9
	CLIP-AD	81.5±2.0	82.7±0.4	82.3±0.5	82.2±0.2
speckle noise	ACR-DSVDD	86.5±2.0	85.8±0.8	86.0±0.4	85.1±0.2
	ACR-BCE	85.9±1.7	86.4±0.4	85.7±0.6	85.4±0.4
	ResNet152-I	75.8±2.8	75.8±0.4	75.1±0.4	72.9±0.5
	ResNet152-II	61.8±2.8	61.0±1.0	61.0±0.9	59.8±0.3
	OC-MAML (1-shot)	52.2±2.7	52.8±1.2	53.5±1.2	53.7±0.4
	CLIP-AD	84.6±1.6	83.7±0.4	84.1±0.4	84.2±0.3
gaussian blur	ACR-DSVDD	88.5±1.1	88.5±0.7	88.7±0.4	88.6±0.3
	ACR-BCE	85.6±1.3	85.0±0.6	85.0±0.9	84.7±0.5
	ResNet152-I	85.2±1.5	83.7±1.0	82.9±0.7	80.9±0.3
	ResNet152-II	64.9±1.5	65.3±1.2	64.0±0.9	62.7±0.4
	OC-MAML (1-shot)	55.6±3.6	56.6±0.6	56.8±1.1	57.6±0.6
	CLIP-AD	91.9±0.8	92.7±0.5	92.1±0.5	92.3±0.2
defocus blur	ACR-DSVDD	89.7±1.8	89.5±0.8	89.1±0.3	89.2±0.3
	ACR-BCE	86.5±1.3	86.5±0.6	86.3±0.3	85.9±0.4
	ResNet152-I	85.9±2.3	85.5±0.8	83.8±0.6	82.4±0.3
	ResNet152-II	66.0±1.8	65.4±1.1	63.7±0.6	63.2±0.3
	OC-MAML (1-shot)	53.5±2.5	51.7±1.7	54.0±1.8	54.7±0.7
	CLIP-AD	93.1±1.4	92.9±0.3	92.8±0.3	92.8±0.2
glass blur	ACR-DSVDD	87.0±2.1	87.9±0.4	87.7±0.4	87.6±0.2
	ACR-BCE	85.4±1.0	86.1±0.4	86.4±0.4	86.1±0.3
	ResNet152-I	80.3±2.5	78.7±0.8	78.0±0.6	75.6±0.4
	ResNet152-II	63.9±2.2	63.0±1.7	63.6±0.5	62.2±0.4
	OC-MAML (1-shot)	52.8±1.9	53.1±1.7	53.9±0.9	53.7±1.4
	CLIP-AD	85.4±0.5	85.0±1.1	84.2±0.7	84.4±0.3
motion blur	ACR-DSVDD	89.2±0.4	89.6±0.8	89.1±0.5	88.6±0.5
	ACR-BCE	86.3±1.9	85.3±1.0	85.7±0.2	84.9±0.2
	ResNet152-I	84.3±1.3	83.4±1.3	82.0±0.5	80.4±0.3
	ResNet152-II	66.6±3.1	64.8±1.2	63.4±0.6	62.4±0.3
	OC-MAML (1-shot)	50.5±3.1	52.3±1.6	53.1±0.9	53.6±0.7
	CLIP-AD	91.8±1.4	92.9±0.4	92.7±0.3	92.8±0.3
zoom blur	ACR-DSVDD	90.3±1.7	89.6±0.7	89.8±0.4	89.4±0.3
	ACR-BCE	87.5±1.8	86.5±1.0	86.4±0.2	86.4±0.3
	ResNet152-I	86.9±1.3	87.2±0.3	86.3±0.3	84.6±0.2
	ResNet152-II	65.2±1.1	65.7±0.7	66.1±0.3	64.2±0.4
	OC-MAML (1-shot)	50.6±3.2	53.8±0.8	53.7±1.4	54.2±0.4
	CLIP-AD	94.2±1.4	94.4±0.3	94.3±0.3	93.9±0.3
snow	ACR-DSVDD	87.7±1.2	87.7±1.0	87.6±0.4	87.4±0.3
	ACR-BCE	84.4±2.6	85.5±0.8	85.5±0.6	84.4±0.0
	ResNet152-I	85.8±1.4	84.8±0.6	83.7±0.8	81.9±0.3
	ResNet152-II	67.1±1.9	65.6±0.9	64.5±0.4	63.3±0.8
	OC-MAML (1-shot)	56.7±4.5	54.5±1.8	56.8±0.6	57.3±0.2
	CLIP-AD	91.7±0.8	92.9±0.4	93.3±0.2	93.2±0.2
fog	ACR-DSVDD	86.2±1.8	85.2±0.6	85.4±0.9	85.0±0.2
	ACR-BCE	78.8±2.7	77.7±0.5	77.3±0.7	77.2±0.6
	ResNet152-I	76.4±1.8	76.9±0.6	74.8±1.0	73.0±0.9
	ResNet152-II	64.5±2.1	62.9±0.8	62.5±0.4	61.0±0.5
	OC-MAML (1-shot)	51.9±3.6	52.9±0.9	53.4±0.6	53.7±0.2
	CLIP-AD	91.9±0.8	92.3±0.5	92.2±0.4	92.3±0.3
frost	ACR-DSVDD	88.2±1.5	88.0±0.9	87.4±0.6	87.2±0.3
	ACR-BCE	83.2±1.4	84.1±1.2	84.6±0.6	83.7±0.4
	ResNet152-I	85.9±1.6	85.5±0.5	83.8±0.8	81.4±0.5
	ResNet152-II	63.0±1.0	63.2±0.5	62.7±1.3	61.7±0.3
	OC-MAML (1-shot)	52.8±1.3	52.4±2.0	53.6±0.7	53.2±1.1
	CLIP-AD	92.9±0.6	93.1±0.2	93.6±0.3	93.2±0.2
brightness	ACR-DSVDD	90.0±1.5	89.5±0.9	89.6±0.4	89.9±0.2
	ACR-BCE	86.7±1.3	87.8±0.7	87.1±0.8	87.2±0.4
	ResNet152-I	90.7±0.9	90.8±0.5	89.7±0.3	88.1±0.3
	ResNet152-II	67.6±2.1	69.8±0.4	68.2±1.0	67.0±0.5
	OC-MAML (1-shot)	53.6±1.1	56.8±1.5	56.2±0.7	56.8±0.5
	CLIP-AD	94.6±0.4	95.6±0.3	95.4±0.3	95.3±0.2
spatter	ACR-DSVDD	88.1±1.5	89.2±0.6	89.0±0.6	88.7±0.1
	ACR-BCE	86.2±2.3	87.7±0.3	87.2±0.6	87.3±0.3
	ResNet152-I	90.6±1.2	90.2±0.5	89.8±0.3	87.9±0.3
	ResNet152-II	68.7±1.7	67.6±0.9	66.0±0.9	65.2±0.4
	OC-MAML (1-shot)	53.6±2.7	55.6±1.1	56.1±0.7	53.6±1.5
	CLIP-AD	94.7±0.6	95.2±0.4	95.1±0.2	95.0±0.3
saturate	ACR-DSVDD	88.1±2.1	87.1±0.7	87.1±0.5	85.8±0.4
	ACR-BCE	86.8±2.0	86.1±0.8	86.0±0.6	85.3±0.3
	ResNet152-I	90.4±0.9	89.7±0.7	89.2±0.5	87.4±0.2
	ResNet152-II	67.7±1.8	67.7±1.4	67.4±0.8	65.9±0.3
	OC-MAML (1-shot)	55.6±2.4	53.5±0.9	55.1±0.8	54.1±1.2
	CLIP-AD	94.7±0.8	94.7±0.2	95.0±0.1	95.1±0.2
contrast	ACR-DSVDD	76.4±1.8	75.1±1.8	74.9±0.5	74.5±0.4
	ACR-BCE	67.6±2.0	66.7±0.8	67.8±0.7	66.9±0.3
	ResNet152-I	76.1±1.6	77.0±0.8	75.2±0.3	73.5±0.2
	ResNet152-II	61.3±0.9	61.3±1.2	60.2±0.5	59.3±0.5
	OC-MAML (1-shot)	54.6±3.7	54.0±0.3	53.1±1.2	54.1±1.0
	CLIP-AD	89.3±1.8	88.9±0.5	88.3±0.4	88.8±0.2
elastic transform	ACR-DSVDD	90.8±1.9	89.3±0.7	90.0±0.4	89.3±0.3
	ACR-BCE	87.6±1.0	86.7±0.8	87.4±0.6	87.2±0.4
	ResNet152-I	82.4±2.4	80.9±0.4	80.0±0.9	78.4±0.2
	ResNet152-II	65.6±2.3	65.2±0.6	63.9±0.7	62.0±0.3
	OC-MAML (1-shot)	52.5±3.9	54.3±1.2	54.4±1.2	54.7±0.8
	CLIP-AD	89.1±1.1	90.0±0.3	89.4±0.5	89.4±0.3
pixelate	ACR-DSVDD	91.7±0.5	91.1±0.6	90.8±0.6	90.7±0.2
	ACR-BCE	89.6±1.9	89.9±0.6	89.7±0.1	89.8±0.3
	ResNet152-I	82.5±1.6	83.2±1.3	82.5±0.7	80.1±0.3
	ResNet152-II	66.4±1.5	65.6±0.6	64.9±0.6	63.8±0.3
	OC-MAML (1-shot)	56.4±3.8	55.8±0.9	56.4±0.7	57.0±0.9
	CLIP-AD	86.7±0.3	86.7±0.7	86.9±0.3	86.7±0.3
jpeg compression	ACR-DSVDD	89.8±1.4	91.0±0.5	90.5±0.7	90.4±0.3
	ACR-BCE	89.1±1.2	88.8±0.8	89.1±0.5	88.6±0.3
	ResNet152-I	84.7±2.1	85.8±1.1	84.4±0.8	82.7±0.2
	ResNet152-II	62.9±2.4	63.9±0.8	63.0±0.8	61.3±0.8
	OC-MAML (1-shot)	52.0±2.7	55.6±0.9	56.4±1.1	57.2±1.8
	CLIP-AD	89.8±1.9	87.7±0.1	88.3±0.3	88.5±0.3

Table D.6: AUC (%) with standard deviation for anomaly detection on non-natural images: Omniglot, MNIST, and OrganA. ACR with both backbone models outperforms all baselines on all datasets. In comparison, CLIP-AD performs much worse on non-natural images.

	MNIST			Omniglot		
	1%	5%	10%	5%	10%	20%
ADIB [Deecke et al., 2021]	50.4±2.0	49.4±1.7	49.4±2.0	50.8±1.7	49.5±0.6	49.7±0.4
ResNet152-I [He et al., 2016]	87.2±1.3	84.2±0.2	80.9±0.2	96.4±0.4	95.5±0.3	94.3±0.2
ResNet152-II [He et al., 2016]	80.0±1.9	78.4±1.5	74.9±0.3	88.1±0.8	86.7±0.5	84.4±0.6
OC-MAML [Frikha et al., 2021]	83.7±3.5	86.0±2.3	86.4±2.8	98.6±0.3	98.4±0.2	98.5±0.1
CLIP-AD [Liznerski et al., 2022]	53.9±1.4	53.7±0.9	53.9±0.8	N/A	N/A	N/A
ACR-DSVDD	**91.9±0.8**	**90.4±0.2**	**88.8±0.2**	**99.1±0.2**	**99.1±0.2**	**99.2±0.0**
ACR-BCE	88.7±0.6	87.8±0.4	86.5±0.3	98.5±0.2	**98.9±0.1**	**99.1±0.1**

Table D.7: ACR-DSVDD's pixel-level (segmentation) and image-level (classification) AUCROCs (%) on MVTec-AD.

	Pixel-level	Image-level
Bottle	95.8±0.5	99.5±0.2
Cable	87.1±1.1	72.2±1.9
Capsule	95.2±0.9	78.1±1.2
Carpet	97.8±0.4	99.8±0.2
Grid	90.4±1.4	83.8±3.0
Hazelnut	92.3±0.7	79.2±0.9
Leather	98.6±0.1	100.0±0.0
Metal-nut	79.5±2.5	75.6±5.4
Pill	93.0±1.3	72.2±3.2
Screw	86.7±0.9	48.5±3.1
Tile	90.3±0.9	98.4±0.3
Toothbrush	98.2±0.1	97.0±0.9
Transistor	97.2±0.1	93.1±1.1
Wood	89.2±1.1	98.2±0.8
Zipper	95.8±0.9	90.7±0.9
Average	92.5±0.2	85.8±0.6

Table D.8: ACR-DSVDD's pixel-level (segmentation) and image-level (classification) AUCROCs (%) on MVTec-AD. The model uses images from other classes as synthetic anomalies during training.

	Pixel-level	Image-level
Bottle	94.5	98.6
Cable	88.1	64.5
Capsule	90.1	70.8
Carpet	97.5	99.5
Grid	74.8	97.9
Hazelnut	84.3	61.9
Leather	97.5	99.1
Metal-nut	67.5	54.2
Pill	89.0	66.8
Screw	76.6	53.2
Tile	90.0	97.6
Toothbrush	93.5	80.8
Transistor	95.2	91.4
Wood	88.1	97.2
Zipper	78.6	79.7
Average	87.0	78.8

Table D.9: AUC (%) with standard deviation for anomaly detection on Malware [Huynh et al., 2017]. ACR-NTL achieves the best results on various anomaly ratios.

	1%	5%	10%	20%
OC-SVM	19.5±5.6	20.5±1.4	20.3±0.9	20.3±0.8
IForest	22.8±2.9	22.9±1.2	23.3±0.6	23.4±0.8
LOF	22.3±4.9	23.2±1.8	23.3±1.3	23.2±0.4
KNN	21.6±6.3	22.5±1.6	22.7±0.9	22.6±0.9
DSVDD	25.4±3.3	27.4±1.7	28.9±0.9	28.3±0.8
AE	48.8±2.4	49.1±1.2	49.4±0.6	49.3±0.5
LUNAR	23.1±4.5	23.8±1.2	24.1±0.7	24.2±0.6
ICL	83.5±1.9	81.0±1.0	82.9±0.8	83.1±0.9
NTL	25.9±4.8	25.4±1.3	24.5±1.3	25.0±0.8
ACR-DSVDD	73.1±2.8	69.5±3.3	69.4±3.3	66.4±4.0
ACR-NTL	**85.0±1.3**	**84.5±0.8**	**85.1±1.2**	**84.0±0.8**

Appendix E

Chapter 6

E.1 Structured Variational Inference

According to the main paper, we consider the generative model $p(\mathbf{x}_t, \mathbf{z}_t, s_t|\mathbf{x}_{1:t-1}, s_{1:t-1}) = p(s_t)p(\mathbf{z}_t|s_t; \boldsymbol{\tau}_t)p(\mathbf{x}_t|\mathbf{z}_t)$ at time step t, where the dependence on $\mathbf{x}_{1:t-1}, s_{1:t-1}$ is contained in $\boldsymbol{\tau}_t$. Upon observing the data $\mathbf{x}_t$, both $\mathbf{z}_t$ and s_t are inferred. However, exact inference is not available due to the intractability of the marginal likelihood $p(\mathbf{x}_t|s_{1:t}, \mathbf{x}_{1:t-1})$. To tackle this, we utilize structured variational inference for both the latent variables $\mathbf{z}_t$ and the Bernoulli change variable s_t. To this end, we define the joint variational distribution $q(\mathbf{z}_t, s_t) = q(s_t|s_{1:t-1})q(\mathbf{z}_t|s_{1:t})$ as in the main paper. For notational simplicity, we omit the dependence on $s_{1:t-1}$. Then the updating procedure for $q(s_t)$ and $q(\mathbf{z}_t|s_t)$ is obtained by maximizing the ELBO $\mathcal{L}(q)$:

$$q^*(\mathbf{z}_t, s_t) = \underset{q(\mathbf{z}_t, s_t) \in Q}{\arg\max}\, \mathcal{L}(q),$$

$$\mathcal{L}(q) := \mathbb{E}_q[\log p(\mathbf{x}_t, \mathbf{z}_t, s_t; \boldsymbol{\tau}_t) - \log q(\mathbf{z}_t, s_t)].$$

Given the generative models, we can further expand $\mathcal{L}(q)$ to simplify the optimization:

$$\begin{aligned}
\mathcal{L}(q) &= \mathbb{E}_{q(s_t)q(\mathbf{z}_t|s_t)}[\log p(s_t) + \log p(\mathbf{z}_t|s_t;\boldsymbol{\tau}_t) + \log p(\mathbf{x}_t|\mathbf{z}_t) - \log q(s_t) - \log q(\mathbf{z}_t|s_t)] \\
&= \mathbb{E}_{q(s_t)}[\log p(s_t) - \log q(s_t) + \mathbb{E}_{q(\mathbf{z}_t|s_t)}[\log p(\mathbf{z}_t|s_t;\boldsymbol{\tau}_t) + \log p(\mathbf{x}_t|\mathbf{z}_t) - \log q(\mathbf{z}_t|s_t)]] \\
&= \mathbb{E}_{q(s_t)}[\log p(s_t) - \log q(s_t) + \mathbb{E}_{q(\mathbf{z}_t|s_t)}[\log p(\mathbf{x}_t|\mathbf{z}_t)] - \mathrm{KL}(q(\mathbf{z}_t|s_t)||p(\mathbf{z}_t|s_t;\boldsymbol{\tau}_t))] \\
&= \mathbb{E}_{q(s_t)}[\log p(s_t) - \log q(s_t) + \mathcal{L}(q|s_t)] \qquad \text{(E.1)}
\end{aligned}$$

where the second step pushes inside the expectation with respect to $q(\mathbf{z}_t|s_t)$, the third step re-orders the terms, and the final step utilizes the definition of CELBO (Eq. 6.7 in the main paper).

Maximizing Eq. E.1 therefore implies a two-step optimization: first maximize the CELBO $\mathcal{L}(q|s_t)$ to find the optimal $q^*(\mathbf{z}_t|s_t = 1)$ and $q^*(\mathbf{z}_t|s_t = 0)$, then compute the Bernoulli distribution $q^*(s_t)$ by maximizing $\mathcal{L}(q)$ while the CELBOs $\mathcal{L}(q^*|s_t)$ are fixed.

While $q^*(\mathbf{z}_t|s_t)$ typically needs to be inferred by black box variational inference [Ranganath et al., 2014, Kingma and Welling, 2014, Zhang et al., 2018], the optimal $q^*(s_t)$ has a closed-form solution and bears resemblance to the exact inference counterpart (Eq. 6.4 in the main paper). To see this, we assume $\mathcal{L}(q^*|s_t)$ are given and $q(s_t)$ is parameterized by $m \in \mathbb{R}$ (for the Bernoulli distribution). Rewriting Eq. E.1 gives

$$\begin{aligned}
\mathcal{L}(q) = {} & m(\log p(s_t = 1) - \log m + \mathcal{L}(q^*|s_t = 1)) \\
& + (1 - m)(\log p(s_t = 0) - \log(1 - m) + \mathcal{L}(q^*|s_t = 0))
\end{aligned}$$

which is concave since the second derivative is negative. Thus taking the derivative and

setting it to zero leads to the optimal solution of

$$\log \frac{m}{1-m} = \log p(s_t = 1) - \log p(s_t = 0) + \mathcal{L}(q^*|s_t = 1)) - \mathcal{L}(q^*|s_t = 0)),$$
$$m = \sigma(\mathcal{L}(q^*|s_t = 1)) - \mathcal{L}(q^*|s_t = 0)) + \xi_0),$$

which attains the closed-form solution as stated in Eq. 6.6 in the main paper without temperature T.

E.2 Additive vs. Multiplicative Broadening

There are several possible choices for defining an informative prior corresponding to $s_t = 1$. In latent time series models, such as Kalman filters [Kalman, 1960, Bamler and Mandt, 2017], it is common to define a linear transition model $\mathbf{z}_t = A\mathbf{z}_{t-1} + \boldsymbol{\epsilon}_t$ where $\mathbf{z}_{t-1} \sim \mathcal{N}(\mathbf{z}_{t-1}; \boldsymbol{\mu}_{t-1}, \Sigma_{t-1})$ and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \Sigma_n)$. Propagating the posterior at time $t-1$ to the prior at time t results in $\mathbf{z}_t \sim \mathcal{N}(\mathbf{z}_t; A\boldsymbol{\mu}_{t-1}, A\Sigma_{t-1}A^\top + \Sigma_n)$. To simplify the discussion, we set $A = I$ and $\Sigma_n = \sigma_n^2 I$; the same argument also applies for the more general case. Adding a constant noise $\boldsymbol{\epsilon}_t$ results in adding the variance of all variables with a constant σ_n^2. We thus call this convolution scheme *additive broadening*. The problem with such a choice, however, is that the associated information loss is not homogeneously distributed: σ_n^2 ignores the uncertainty in $\mathbf{z}_t$, and dimensions of $\mathbf{z}_t$ with low posterior uncertainty lose more information relative to dimensions of $\mathbf{z}_t$ that are already uncertain. We found that this scheme deteriorates the learning signal.

We therefore consider *multiplicative broadening* (or *relative broadening* since the associated information loss depends on the original variance) as *tempering* described in the main paper, resulting in $p(\mathbf{z}_t|s_t, \boldsymbol{\tau}_t) \propto p(\mathbf{z}_{t-1}|\mathbf{x}_{1:t-1}, s_{1:t-1})^\beta$ for $\beta > 0$. For a Gaussian distribution, the resulting variance scales the original variance with $\frac{1}{\beta}$. In practice, we found relative

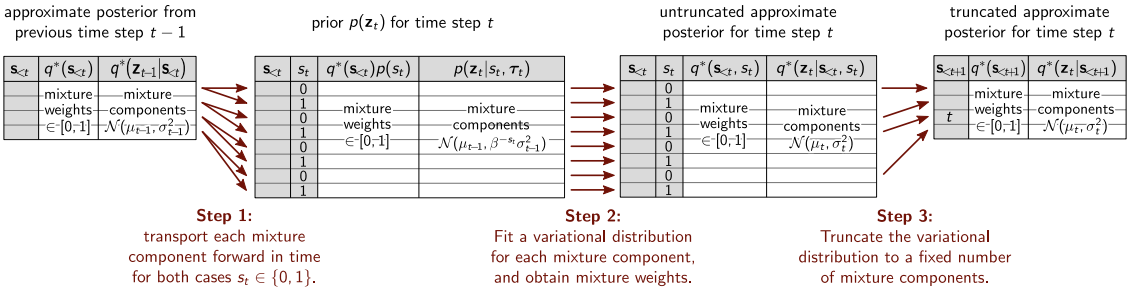


Figure E.1: Conditional probability table of variational beam search

or multiplicative broadening to perform much better and robustly than additive posterior broadening. Since tempering broadens the posterior non-locally, this scheme does not possess a continuous latent time series interpretation [1].

E.3 Details on "Shy" Variational Greedy Search and Variational Beam Search

"Shy" Variational Greedy Search. As illustrated in Fig. 6.1 in the main text, one obtains better interpretation if one outputs the variational parameters μ_t and σ_t at the end of a segment of constant $\mathbf{z}_t$. More precisely, when the algorithm detects a change point $s_t = 1$, it outputs the variational parameters μ_{t-1} and σ_{t-1} from just before the detected change point t. These parameters define a variational distribution that has been fitted, in an iterative way, to all data points since the preceding detected change point. We call this the "shy" variant of the variational greedy search algorithm, because this variant quietly iterates over the data and only outputs a new fit when it is as certain about it as it will ever be. The red lines and regions in Fig. 6.1 (a) in the main paper illustrate means and standard deviations outputted by the "shy" variant of variational greedy search.

[1] This means that it is impossible to specify a conditional distribution $p(\mathbf{z}_t|\mathbf{z}_{t-1})$ that corresponds to relative broadening.

Algorithm 4: Variational Beam Search

Require: task set $\{\mathbf{x}_t\}_1^T$; beam size K; prior log-odds ξ_0; conditional ELBO temperature T
Ensure: approximate posterior distributions $\{q^*(s_{1:t}), q^*(\mathbf{z}_t|s_{1:t})\}_1^T$

$q^*(\mathbf{z}_1) = \arg\max \mathbb{E}_q[\log p(\mathbf{x}_1|\mathbf{z}_1)] - \mathrm{KL}(q(\mathbf{z}_1)||p(\mathbf{z}_1))$;
$q^*(s_1 = 0) := 1$; $q^*(\mathbf{z}_1|s_1) := q^*(\mathbf{z}_1)$; $\mathbb{B} = \{s_1 = 0\}$;
for $t = 1, \cdots, T$ **do**
  $\mathbb{B}' = \{\}$
  for each hypothesis $\mathbf{s}_{<t} \in \mathbb{B}$ **do**
    $p(s_t = 1) := \sigma(\xi_0)$ for random variable $s_t \in \{0, 1\}$
    $\mathbb{B}' := \mathbb{B}' \cup \{(\mathbf{s}_{<t}, s_t = 0), (\mathbf{s}_{<t}, s_t = 1)\}$;
    compute the task t's prior $p(\mathbf{z}_t|s_t, \boldsymbol{\tau}_t)$ (Eq. 6.3);
    perform structured variational inference (Eq. 6.7 and Eq. 6.6) given observation $\mathbf{x}_t$, resulting in $q^*(s_t, \mathbf{z}_t|\mathbf{s}_{<t}) = q^*(s_t|\mathbf{s}_{<t})q^*(\mathbf{z}_t|\mathbf{s}_{<t+1})$ where $q^*(\mathbf{z}_t|\mathbf{s}_{<t+1}$ is stored as output $q^*(\mathbf{z}_t|s_{1:t})$;
    approximate new hypotheses' posterior probability $p(s_{1:t}|\mathbf{x}_{1:t}) \approx q^*(\mathbf{s}_{<t}, s_t) = q^*(\mathbf{s}_{<t})q^*(s_t|\mathbf{s}_{<t})$;
  end for
  $\mathbb{B} :=$ `diverse_truncation`$(\mathbb{B}', q^*(\mathbf{s}_{<t}, s_t))$;
  normalize $q^*(\mathbf{s}_{<t}, s_t)$ where $(\mathbf{s}_{<t}, s_t) \in \mathbb{B}$;
end for

We applied this "Shy" variant to our illustrative example (Section 6.4.1) and unsupervised learning experiments (Section 6.4.5).

Variational Beam Search. As follows, we present a more detailed explanation of the variational beam search procedure mentioned in Section 6.3.3 of the main paper. Our beam search procedure defines an effective way to search for potential hypotheses with regards to sequences of inferred change points. The procedure is completely defined by detailing three sequential steps, that when executed, take a set of hypotheses found at time step $t-1$ and transform them into the resulting set of likely hypotheses for time step t that have appropriately accounted for the new data seen at t. The red arrows in Figure E.1 illustrate these three steps for beam search with a beam size of $K = 4$.

In Figure E.1, each of the three steps maps a table of considered histories to a new table. Each table defines a mixture of Gaussian distributions where each mixture component corresponds

to a different history and is represented by a different row in the table. We start on the left with the (truncated) variational distribution $q^*(\mathbf{z}_{t-1})$ from the previous time step, which is a mixture over $K = 4$ Gaussian distributions. Each mixture component (row in the table) is labeled by a 0-1 vector $\mathbf{s}_{<t} \equiv (s_1, \cdots, s_{t-1})$ of the change variable values according to that history. Each mixture component $\mathbf{s}_{<t}$ further has a mixture weight $q^*(\mathbf{s}_{<t}) \in [0, 1]$, a mean, and a standard deviation.

We then obtain a prior for time step t by transporting each mixture component of $q^*(\mathbf{z}_{t-1})$ forward in time via the broadening functional ("Step 1" in the above figure). The prior $p(\mathbf{z}_t)$ (second table in the figure) is a mixture of $2K$ Gaussian distributions because each previous history splits into two new ones for the two potential cases $s_t \in \{0, 1\}$. The label for each mixture component (table row) is a new vector $(\mathbf{s}_{<t}, s_t)$ or $\mathbf{s}_{<t+1}$, appending s_t to the tail of $\mathbf{s}_{<t}$.

"Step 2" in the above figure takes the data $\mathbf{x}_t$ and fits a variational distribution $q^*(\mathbf{z}_t)$ that is also a mixture of $2K$ Gaussian distributions. To learn the variational distribution, we (i) numerically fit each mixture component $q(\mathbf{z}_t|\mathbf{s}_{<t}, s_t)$ individually, using the corresponding mixture component of $p(\mathbf{z}_t)$ as the prior; (ii) evaluate (or estimate) the CELBO of each fitted mixture component, conditioned on $(\mathbf{s}_{<t}, s_t)$; (iii) compute the approximate posterior probability $q^*(s_t|\mathbf{s}_{<t})$ of each mixture component, in the presence of the CELBOs; and (iv) obtain the mixture weight equal to the posterior probability over $(\mathbf{s}_{<t}, s_t)$, i.e., $p(s_{1:t}|\mathbf{x}_{1:t})$, best approximated by $q^*(\mathbf{s}_{<t})q^*(s_t|\mathbf{s}_{<t})$.

"Step 3" in the above figure truncates the variational distribution by discarding K of the $2K$ mixture components. The truncation scheme can be either the "vanilla" beam search or diversified beam search outlined in the main paper. The truncated variational distribution $q_t(\mathbf{z}_t)$ is again a mixture of only K Gaussian distributions, and it can thus be used for subsequent update steps, i.e., from t to $t + 1$.

The pseudocode is listed in Algo 4.

E.4 Online Bayesian Linear Regression with Variational Beam Search

This section will derive the analytical solution of online updates for both Bayesian linear regression and the probability of change points. We consider Gaussian prior distributions for weights. The online update of the posterior distribution is straightforward in the natural parameter space, where the update is analytic given the sufficient statistics of the observations. If we further allow to temper the weights' distributions with a fixed temperature β, then this corresponds to multiplying each element in the precision matrix by β. We applied this algorithm to the linear regression experiments in Section 6.4.3. For unified names, we still use the word "variational" even though the solutions are analytical.

E.4.1 Variational Continual Learning for Online Linear Regression

Let's start with assuming a generative model at time t:

$$\begin{aligned} &\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma), \\ &y_t = \boldsymbol{\theta}^\top \mathbf{x}_t + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_n^2), \end{aligned} \tag{E.2}$$

and the noise ϵ is constant over time.

The posterior distribution of $\boldsymbol{\theta}$ is of interest, which is Gaussian distributed since both the likelihood and prior are Gaussian. To get an online recursion for $\boldsymbol{\theta}$'s posterior distribution

over time, we consider the natural parameterization. The prior distribution under this parameterization is

$$\begin{aligned} p(\boldsymbol{\theta}) &= \frac{1}{Z}\exp\left(-\frac{(\boldsymbol{\theta}-\boldsymbol{\mu})^\top\Sigma^{-1}(\boldsymbol{\theta}-\boldsymbol{\mu})}{2}\right) \\ &= \frac{1}{Z}\exp\left(-\frac{1}{2}\boldsymbol{\theta}^\top\Sigma^{-1}\boldsymbol{\theta}+\boldsymbol{\theta}^\top\Sigma^{-1}\boldsymbol{\mu}\right) \\ &= \frac{1}{Z}\exp\left(-\frac{1}{2}\boldsymbol{\theta}^\top\Lambda\boldsymbol{\theta}+\boldsymbol{\theta}^\top\boldsymbol{\eta}\right) \end{aligned}$$

where $\Lambda = \Sigma^{-1}, \boldsymbol{\eta} = \Sigma^{-1}\boldsymbol{\mu}$ are the natural parameters and the terms unrelated to $\boldsymbol{\theta}$ are absorbed into the normalizer Z.

Following the same parameterization, the posterior distribution can be written

$$\begin{aligned} p(\boldsymbol{\theta}|\mathbf{x}_t, y_t) &\propto p(\boldsymbol{\theta})p(y_t|\mathbf{x}_t,\boldsymbol{\theta}) \\ &= \frac{1}{Z}\exp\left(-\frac{1}{2}\boldsymbol{\theta}^\top\Lambda\boldsymbol{\theta}+\boldsymbol{\theta}^\top\boldsymbol{\eta}-\frac{1}{2}\sigma_n^{-2}\boldsymbol{\theta}^\top(\mathbf{x}_t\mathbf{x}_t^\top)\boldsymbol{\theta}+\sigma_n^{-2}y_t\boldsymbol{\theta}^\top\mathbf{x}_t\right) \\ &= \frac{1}{Z}\exp\left(-\frac{1}{2}\boldsymbol{\theta}^\top(\Lambda+\sigma_n^{-2}\mathbf{x}_t\mathbf{x}_t^\top)\boldsymbol{\theta}+\boldsymbol{\theta}^\top(\boldsymbol{\eta}+\sigma_n^{-2}y_t\mathbf{x}_t)\right). \end{aligned}$$

Thus we get the recursion over the natural parameters

$$\begin{aligned} \Lambda' &= \Lambda + \sigma_n^{-2}\mathbf{x}_t\mathbf{x}_t^\top, \\ \boldsymbol{\eta}' &= \boldsymbol{\eta} + \sigma_n^{-2}y_t\mathbf{x}_t, \end{aligned}$$

from which the posterior mean and covariance can be solved.

E.4.2 Prediction and Marginal Likelihood

We can get the posterior predictive distribution for a new input $\mathbf{x}_*$ through inspecting Eq. E.2 and utilizing the linear properties of Gaussian. Assuming the generative model as specified

above, we replace $\mathbf{x}_t$ with $\mathbf{x}_*$ in Eq. E.2. Since $\boldsymbol{\theta}$ is Gaussian distributed, by its linear property, $\mathbf{x}_*^\top \boldsymbol{\theta}$ conforms to $\mathcal{N}(\mathbf{x}_*^\top \boldsymbol{\theta}; \mathbf{x}_*^\top \boldsymbol{\mu}, \mathbf{x}_*^\top \Sigma \mathbf{x}_*)$. Then the addition of two independent Gaussian results in $y_* \sim \mathcal{N}(y_*; \mathbf{x}_*^\top \boldsymbol{\mu}, \sigma_n^2 + \mathbf{x}_*^\top \Sigma \mathbf{x}_*)$.

The marginal likelihood shares this same form with the posterior predictive distribution, with a potentially different pair of sample $(\mathbf{x}, y)$. To see this, given a prior distribution $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \Sigma)$, then the marginal likelihood of $y|\mathbf{x}$ is

$$\begin{aligned} p(y|\mathbf{x}; \boldsymbol{\mu}, \Sigma, \sigma_n) &= \int p(y|\mathbf{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta}; \boldsymbol{\mu}, \Sigma) d\boldsymbol{\theta} \\ &= \mathcal{N}(y; \mathbf{x}^\top \boldsymbol{\mu}, \sigma_n^2 + \mathbf{x}^\top \Sigma \mathbf{x}) \end{aligned} \tag{E.3}$$

with σ_n^2 being the noise variance. Note that in variational inference with an intractable marginal likelihood (not like the linear regression here), this is the approximated objective (Evidence Lower Bound (ELBO), indeed) we aim to maximize.

Computation of the Covariance Matrix Since we parameterize the precision matrix instead of the covariance matrix, the variance of the new test sample requires to take the inverse of the precision matrix. In order to do this, we employ the eigendecomposition of the precision matrix and re-assemble to the covariance matrix through inverting the eigenvalues. A better approach is to apply the Sherman-Morrison formula for the rank one update[2], which can reduce the computation from $O(n^3)$ to $O(n^2)$.

Logistic Normal Model If we are modeling the log-odds by a Bayesian linear regression, then we need to map the log-odds to the interval $[0, 1]$ by the sigmoid function, to make it a valid probability. Specifically, suppose $a = \mathcal{N}(a; \mu_a, \sigma_a^2)$ and $y = \sigma(a)$ (note we abuse σ by variances and functions, but it is clear from the context and the subscripts) where $\sigma(\cdot)$

[2] `https://en.wikipedia.org/wiki/Sherman%E2%80%93Morrison_formula`

is a logistic sigmoid function. Then y has a logistic normal distribution. Given $p(y)$, we can make decisions for the value of y. There are three details that worth noting. First is from the non-linear mapping of $\sigma(\cdot)$. One special property of $p(y)$ is that $p(y)$ can be bimodal if the base variance or σ_a is large. A consequence is that the mode of $p(a)$ does not necessarily correspond to $p(y)$'s mode and $\mathbb{E}[y] \neq \sigma(\mathbb{E}[a])$. Second is for the binary classification: the decision boundary of y, i.e., 0.5, is consistent with the one of x, i.e., 0, for decisions either by $\mathbb{E}[y]$ or by $\mathbb{E}[a]$. See Rusmassen and Williams [2005] (Section 3.4) and Bishop et al. [1995] (Section 10.3). Third, if our loss function for decision making is the absolute error, then the best prediction is the median of $y = \sigma(\hat{a})$ [Friedman et al., 2001] where $\hat{a}$ is the median of a. This follows from the monotonicity of $\sigma(\cdot)$ that does not change the order statistics.

E.4.3 Inference over the Change Variable

To infer the posterior distribution of s_t given observations $(\mathbf{x}_{1:t}, y_{1:t})$, we apply Bayes' theorem to infer the posterior log-odds as in the main paper

$$\log\left(\frac{p(s_t = 1|\mathbf{x}_{1:t}, y_{1:t}, s_{1:t-1})}{p(s_t = 0|\mathbf{x}_{1:t}, y_{1:t}, s_{1:t-1})}\right) = \frac{\log(p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, y_{1:t}, s_t = 1, s_{1:t-1}))}{\log(p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, y_{1:t}, s_t = 0), s_{1:t-1})} + \xi_0$$

where $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, y_{1:t}, s_t)$ is exactly Eq. E.3 but has different parameter values dictated by s_t and β.

In the next part, we show the resulting distribution of the broadening operation.

Tempering a Multivariate Gaussian

We will show the tempering operation of a multivariate Gaussian corresponds to multiplying each element in the precision matrix by the fixed temperature, a simple form in the natural space.

Suppose we allow to temper / broaden the $\boldsymbol{\theta}$'s multivariate Gaussian distribution before the next time step, accommodating more evidence. Let the broadening constant or temperature be $\beta \in (0, 1]$. We derive how β affects the multivariate Gaussian precision.

Write the tempering explicitly,

$$\begin{aligned} p(\boldsymbol{\theta})^{\beta} &= \frac{1}{Z} \exp\left(-\frac{1}{2}\beta(\boldsymbol{\theta}-\boldsymbol{\mu})^{\top}\Lambda(\boldsymbol{\theta}-\boldsymbol{\mu})\right) \\ &= \frac{1}{Z} \exp\left(-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\mu})^{\top}\Lambda_{\beta}(\boldsymbol{\theta}-\boldsymbol{\mu})\right) \end{aligned}$$

among which we are interested in the relationship between Λ_{β} and Λ and β. To this end, re-write the quadratic form in the summation

$$\begin{aligned} & \beta(\boldsymbol{\theta}-\boldsymbol{\mu})^{\top}\Lambda(\boldsymbol{\theta}-\boldsymbol{\mu}) \\ =& \sum_{i,j} \beta\Lambda_{ij}(\boldsymbol{\theta}_i-\boldsymbol{\mu}_i)(\boldsymbol{\theta}_j-\boldsymbol{\mu}_j) \\ =& \sum_{i,j} \Lambda_{\beta,ij}(\boldsymbol{\theta}_i-\boldsymbol{\mu}_i)(\boldsymbol{\theta}_j-\boldsymbol{\mu}_j) \end{aligned}$$

where we can identify an element-wise relation: for all possible i, j

$$\Lambda_{\beta,ij} = \beta\Lambda_{ij}.$$

Prediction

As above, we are interested in the posterior predictive distribution for a new test sample $(\mathbf{x}_*, y_*)$ after absorbing evidence. Let's denote the parameters of $\boldsymbol{\theta}$'s posterior distributions by $\boldsymbol{\mu}_{s_{1:t}}$ and $\Sigma_{s_{1:t}}$, where the dependence over $s_{1:t}$ is made explicit. We then make posterior

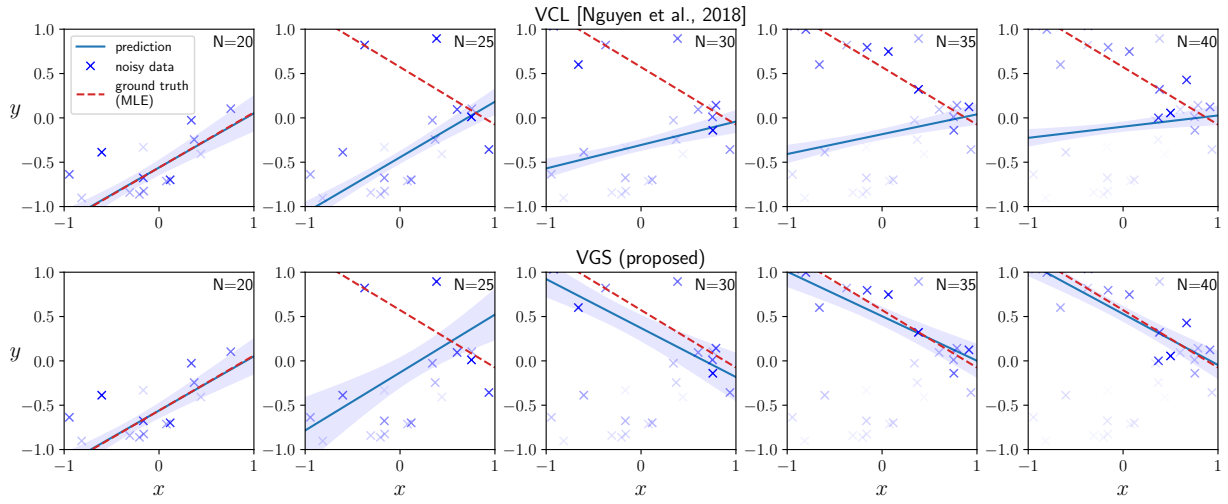


Figure E.2: 1D online Bayesian linear regression with distribution shift. More recent samples are colored darker. Due to the catastrophic remembering, VCL fails to adapt to new observations.

predictions with each component

$$p(y_*|\mathbf{x}_*, \mathbf{x}, y, s_{1:t}) = \mathcal{N}(y_*; \mathbf{x}_*^\top \boldsymbol{\mu}_{s_{1:t}}, \sigma_n^2 + \mathbf{x}_*^\top \Sigma_{s_{1:t}} \mathbf{x}_*).$$

E.5 Visualization of Catastrophic Remembering Effects

In order to demonstrate the effect of catastrophic remembering, we consider a simple linear regression model. We will see that, when the data distribution changes, a Bayesian online learning framework becomes quickly overconfident and unable to adjust to the changing data distribution. On the other hand, with tempering, variational greedy search (VGS) can partially forget the previous knowledge and then adapt to the shifted distribution.

Data Generating Process We generated the samples by the following generative model:

$$x \sim \text{Unif}(-1, 1),$$
$$y \sim \mathcal{N}(f(x), 0.1^2)$$

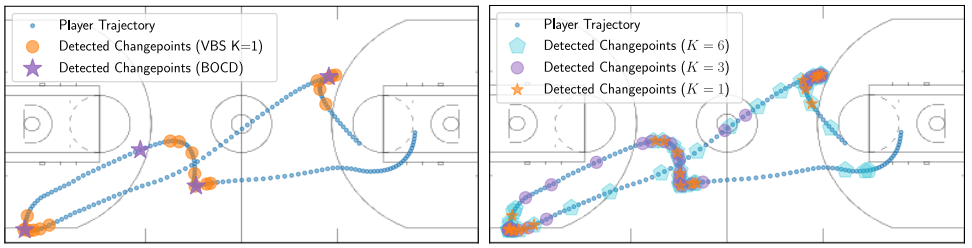
where $f(x)$ equals $f_1(x) = 0.7x - 0.5$ or $f_2(x) = -0.7x + 0.5$. In this experiments, we sampled the first 20 points from f_1 and the remaining 20 points from f_2.

Model Parameters We applied the Bayes updates mentioned in Section E.4 to do inference over the slope and intercept. We set the initial priors of the weights to be standard Gaussian and the observation noise σ_n^2 to be the true scale, 0.1. This setting is enough for VCL.

For VGS, we set the same noise variance $\sigma_n^2 = 0.1$. For the method-specific parameters, we set $\xi_0 = \log(0.35/(1 - 0.35))$ and $\beta = 1/3.5$.

We plotted the noise-free posterior predictive distribution for both VCL and VGS. That is, let $f_*(x)$ be the fitted function, we plotted $p(f_*(x_t)|x_t, \mathcal{D}_{1:t-1}) = \int p(f_*(x_t)|x_t, \mathbf{w}, \mathcal{D}_{1:t-1})p(\mathbf{w}|\mathcal{D}_{1:t-1})d\mathbf{w}$ where $\mathcal{D}_{1:t-1}$ is the observed samples so far.

Results We first visualized the catastrophic remembering effect through a 1D online Bayesian linear regression task where a distribution shift occurred unknown to the regressor (Fig. E.2). In this setup, noisy data were sampled from two ground truth functions $f_1(x) = 0.7x - 0.5$ and $f_2(x) = -0.7x + 0.5$, where, with constant additive noise, the first 20 samples came from f_1 and the remaining 20 samples were from f_2. The observed sample is presented one by one to the regressor. Before the regression starts, the weights (slope and intercept) were initialized to be standard Gaussian distributions. We experimented two different online regression methods, original online Bayesian linear regression (VCL [Nguyen



Beam Size	LogLike↑
$K = 6$	0.252
$K = 3$	0.227
$K = 1$	0.150

Figure E.3: Changepoints in Basketball player movement tracking. (**Left plot**) Comparisons between VBS and BOCD [Adams and MacKay, 2007]. While BOCD detect change points at sparse, abrupt changes, VBS detects the changes at smooth, gradual changes. (**Right plot**) Ablation study over beam size K for VBS while fixing other parameters. As we increase the beam size, qualitatively different change points are detected and the predictive likelihood improves.

et al., 2018a]) and the proposed variational greedy search (VGS). In Fig. E.2, to show a practical surrogate for the ground truth, we plotted the maximum likelihood estimation (MLE) for each function given the observations. The blue line and the shaded area correspond to the mean and standard deviation of the posterior predictive distribution after observing N samples. As shown in Fig. E.2, both VCL (top) and VGS (bottom) faithfully fit to f_1 after observing the first 20 samples. However, when another 20 new observations are sampled from f_2, VCL shows catastrophic remembering of f_1 and cannot adapt to f_2. VGS, on the other hand, tempers the prior distribution automatically and succeeds in adaptation.

E.6 NBAPlayer: Change Point Detection Comparisons

VBS vs. BOCD We investigated the changepoint detection characteristics of our proposed methods and compared against the BOCD baseline in Fig. E.3 (left). On the shown example trajectory, BOCD detects abrupt change points, corresponding to different plays, a similar phenomenon observed by [Harrison et al., 2020]. However, we argue that it is insufficient and late to identify a player's strategy purpose – it only triggers an alarm after a new play starts. VBS, on the other hand, characterizes the transition phases between plays, triggers an early alarm before the next play starts. It also shows the difference between

BOCD and VBS in changepoint detection: while BOCD only detects abrupt changes, VBS detects gradual changes as well.

Practical Considerations of VBS and BOCD Using our variational inference extensions of BOCD, we can overcome the inference difficulty of non-conjugate models. But, considering practical issues, VBS is better in that the detected change points are easy to read from the binary change variable values $s_{1:t}$ and free from post-processing – a procedure that BOCD has to exercise. BOCD often outputs a sequence of run lengths either online or offline, among which the change points do not always correspond to the time when the most probable run length becomes one. Instead, the run length oftentimes is larger than one when change point happens. Then people have to inspect the run lengths and set a subjective threshold to determine when a change point occurs, which is not data-driven and may incur undesirable detection. For example, in our basketball player tracking experiments, we set the threshold of change points to be 50; VBS, on the other hand, is free from this post-process thresholding and provides multiple plausible, completely data-driven hypotheses of change points.

Ablation Study of Beam Size The right plot and the table in Fig E.3 shows, on the example trajectory, the detected change points and the average log-likelihood as the beam size K changes. When $K = 1$, VBS characterizes the trajectory where the velocity direction changes; when $K = 3$ or 6, it seems that some parts where the velocity value changes are detected. We also observed that the average predictive log-likelihood improves as K increases.

E.7 Experiment Details and Results

In this section, we provide the unstated details of the experiments mentioned in the main paper. These details include but are not limited to hardware infrastructure used to experiment, physical running time, hyperparameter searching, data generating process, evaluation metric, additional results, empirical limitations, and so on. The subsection order corresponds to the experiments order in the main paper. We first provide some limitations of our methods during experiments.

Limitations Our algorithm is theoretically sound. The generality and flexibility renders a great performance in experiments, however, at the expense of taking more time to search the hyperparameters in a relatively large space. Specifically, there are two hyperparameters to tune: ξ_0 and β. The grid search over these two hyperparameters could be slow. When we further take into account the beam size K, it adds more burden in parameter searching. But we give a reference scope to the tuning region where the search should perform. Oftentimes, we use the same parameters across beam sizes.

E.7.1 An Illustrative Example

Data Generating Process To generate Figure 6.2 in the main paper, we used a step-wise function as ground truth, where the step size was 1 and two step positions were chosen randomly. We sampled 30 equally-spaced points with time spacing 1. To get noisy observations, Gaussian noise with standard deviation 0.5 was added to the points.

Model Parameters In this simple one-dimensional model, we used absolute broadening with a Gaussian transition kernel $K(\mathbf{z}_t, \mathbf{z}'_t) = \mathcal{N}(\mathbf{z}_t - \mathbf{z}'_t, D\Delta t)$ where $D = 1.0$ and $\Delta t = 1$. The inference is thus tractable because $p(\mathbf{z}_t|s_t)$ is conditional conjugate to $p(\mathbf{x}_t|\mathbf{z}_t, s_t)$ (and

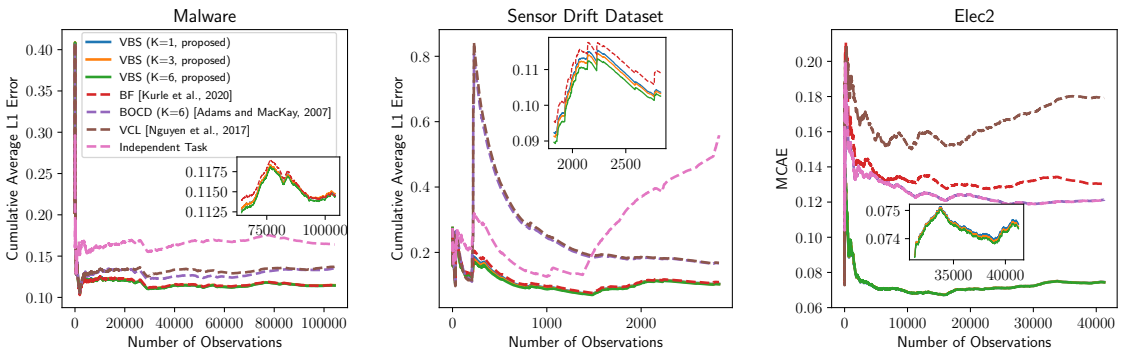


Figure E.4: One-step ahead performance of the online learning experiments. Proposed methods outperform the baseline.

both are Gaussian distributed). We set the prior log-odds ϵ_0 to $\log \frac{p(s_t=1)}{p(s_t=0)}$, where $p(s_t = 1) = 0.1$. We used beam size 2 to do the inference.

E.7.2 Bayesian Linear Regression Experiments

We performed all linear regression experiments on a laptop with 2.6 GHz Intel Core i5 CPU. All models on SensorDrift, Elec2, and NBAPlayer dataset finished running within 5 minutes. Running time on Malware dataset varied: VCL, BF, and Independent task are within 10 minutes; VGS takes about two and half hours; VBS (K=3) takes about six hours; VBS (K=6) takes about 12 hours. BOCD takes similar time with VBS. The main difference between VCL's computation cost and VBS's computation cost lies in the necessarity of inverting the precision matrix into covariance matrix. However, the matrix inverse computation in VBS and BOCD can be substantially reduced from $O(n^3)$ to $O(n^2)$ by the recursion of Sherman–Morrison formula; we will implement this in the future.

Problem Definitions We considered both classification experiments (Malware, Elec2) and regression experiments (SensorDrift, NBAPlayer). The classification datasets have real-value probabilities as targets, permitting to perform regression in log-odds space.

Setup and Evaluation We defined each task to consist of a single observation. Models made predictions on the next observation immediately after finishing learning current task. Models were then evaluated with one-step-ahead absolute error[3], which is then used to compute the mean cumulative absolute error (MCAE) at time t: $\frac{1}{t}\sum_{i=1}^{t}|y_i^* - y_i|$ where y^* is the predicted value and y_i is the ground truth. We further approximated the Gaussian posterior distribution by a point mass centered around the mode. It should be noticed that for linear regression, the Laplace Propagation has the same mode as Variational Continual Learning, and the independent task has the same mode as its Bayesian counterpart.

Results We reported the result of the dominant hypothesis of VBS with large beam size. Fig. E.4 shows MCAE over time for the first three datasets. Our methods remain lower prediction error of all time while baselines are subject to strong fluctuations or inability to adapt to distribution shifts. Another observation is that VBS with different beam sizes performed similarly in this case.

Baseline Hyperparameters

BOCD We only keep the top three or six most probable run length after each time step.

We tuned the hyperparameter λ in the hazard function, or the transition probability. λ^{-1} is searched in $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99\}$.

Malware selects $\lambda^{-1} = 0.3$; Elec2 selects $\lambda^{-1} = 0.9$; SensorDrift selects $\lambda^{-1} = 0.6$; NBAPlayer selects $\lambda^{-1} = 0.99$.

BF We implemented Bayesian Forgetting according to [Kurle et al., 2020].

[3]in probability space for classification tasks; in data space for regression tasks. With the exception of NBAPlayer dataset, we evaluated models with predictive log probability $\frac{1}{t}\sum_{i=2}^{t}\log p(y_i|y_{1:i-1}, x_{1:i})$.

We tuned the hyperparameter β as the forgetting rate such that $p(z_t|D_{t-1}) \propto p_0(z_t)^{1-\beta} q_{t-1}(z_t|D_{t-1})^{\beta}$ where $0 < \beta < 1$. β is searched in $\{0.001, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.97, 0.98, 0.99, 0.995, 0.999\}$.

Malware selects $\beta = 0.999$; Elec2 selects $\beta = 0.98$; SensorDrift selects $\beta = 0.9$; NBAPlayer selects $\beta = 0.9$.

Malware[4]

Dataset There are 107856 programs collected from 2010.11 to 2014.7 in the monthly order. Each program has 482 counting features and a real-valued probability $p \in [0, 1]$ of being malware. This ground truth probability is the proportion of 52 antivirus solutions that label malware. We used the first-month data (2010.11) as the validation dataset and the remaining data as the test dataset. To enable analytic online update, we cast the binary classification problem in the log-odds space and performed Bayesian linear regression. We filled log-odds boundary values to be -5 and 4, corresponding to probability 0 and 1, respectively. Our methods achieved comparable results reported in [Huynh et al., 2017] on this dataset.

Hyperparameters We searched the hyperparameters $\sigma_n^2, \xi_0 = \log \frac{p_0}{1-p_0}$, and β using the validation set. Specifically, we extensively searched $\sigma_n^2 \in \{0.1, 0.2, \cdots, 0.9, 1, 2, \cdots, 10, 20, \cdots, 100\}$, $p_0 \in \{0.5\}$, $\beta^{-1} \in \{1.01, 1.05, 1.1, 1.2, 1.5, 2, 5\}$. On most values the optimization landscape is monotonic and thus the search quickly converges around a local optimizer. Within the local optimizer, we performed the grid search, which focused on $\beta \in [1.05, 1.2]$.

We found all methods favored $\sigma_n^2 = 40$. And for VGS and VBS, the uninformative prior of the change variable $p_0 = 0.5$ was already a suitable one. VGS selected $\beta^{-1} = 1.2$, VBS (K=3) selected $\beta^{-1} = 1.07$, and VBS (K=6) selected $\beta^{-1} = 1.05$. Although searched β^{-1}

[4] `https://archive.ics.uci.edu/ml/datasets/Dynamic+Features+of+VirusShare+Executables`

varies for different beam size, the performance of different beam size in this case, based on our experience, is insensitive to the varying β.

SensorDrift[5]

Dataset We focused on one kind of gas, *Acetaldehyde*, retrieved from the original gas sensor drift dataset [Vergara et al., 2012], which spans 36 months. We formulated an regression problem of predicting the gas concentration level given all the other features. The dataset contains 128 features and 2926 samples in total. We used the first batch data (in the original dataset) as the validation set and the others as the test set. Since the scales of the features vary greatly, we scaled each feature with the sample statistics computed from the validation set, leading to zero mean and unit standard deviation for each feature.

Hyperparameters We found that using a Bayesian forgetting module (instead of tempered posterior module mentioned in the main paper), which corresponds to $s_t = 1$, works better for this dataset. Since we scaled the dataset, we therefore set the hyperparameter $\sigma_n^2 = 1$ for all methods. We searched $\xi_0 = \log \frac{p_0}{1-p_0}$ and β using the validation set. Specifically, we did the grid search for the prior probability of change point $p \in \{0.501, 0.503, 0.505, 0.507, 0.509, 0.511, 521, 0.55, 0.6, 0.7, 0.8, 0.9\}$ and the temperature $\beta \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$. The search procedure selects $p = 0.507$ and $\beta = 0.7$ for all beam size K.

[5] `http://archive.ics.uci.edu/ml/datasets/Gas+Sensor+Array+Drift+Dataset+at+Different+Concentrations`

Elec2[6]

Dataset The dataset contains the electricity price over three years of two Australian states, New South Wales and Victoria [Harries and Wales, 1999]. While the original problem was 0-1 binary classification, we re-produced the targets with real-value probabilities since all necessary information forming the original target is contained in the dataset. Specifically, we re-defined the target to be the probability of the price in New South Wales increasing relative to the price of the last 24 hours. Then we performed linear regression in the log-odds space. We filled log-odds boundary values to be -4 and 4, corresponding to probability 0 and 1, respectively. After removing the first 48 samples (for which we cannot re-produce the targets), we had 45263 samples, and each sample comprised 14 features. The first 4000 samples were used for validation while the others were used for test.

Hyperparameters We searched the hyperparameters $\sigma_n^2, \xi_0 = \log \frac{p_0}{1-p_0}$, and β using the validation set. Specifically, we extensively searched $\sigma_n^2 \in \{0.01, 0.02, \cdots, 0.1, 0.2, \cdots, 1, 2, \cdots, 10, 20, \cdots, 10$
$p_0 \in \{0.5\}$, $\beta^{-1} \in \{1.05, 1.1, 1.2, 1.5, 2, 5\}$.

VCL favored $\sigma_n^2 = 0.01$, and we set this value for all other methods. VGS selected $\beta^{-1} = 1.2$. VBS (K=3) and VBS (K=6) inherited the same β value from VGS.

NBAPlayer[7]

Dataset The original dataset contains part of the game logs of 2015-2016 NBA season in json files. The log records each on-court player's position (in a 2D space) at a rate of 25 Hz. We pre-processed the logs and randomly extracted ten movement trajectories for training set and another ten trajectories for test set. For an instance of the trajectory, we selected

[6] `https://www.openml.org/d/151`
[7] `https://github.com/linouk23/NBA-Player-Movements`

Wesley Matthews's trajectory at the 292th event in the game of Los Angeles Clippers vs. Dallas Mavericks on Nov 11, 2015. The trajectories vary in length and correspond to players' strategic movement. After extracting the trajectories, we fix the data set and then evaluate all methods with it. Specifically, we regress the current position on the immediately previous position–modeling the player's velocity.

Hyperparameters We searched the hyperparameters $\sigma_n^2, \xi_0 = \log \frac{p_0}{1-p_0}$, and β using the training set. Specifically, we searched $\sigma_n^2 \in \{0.001, 0.01, 0.1, 0.5, 1.0, 10., 100.\}$ and $p \in \{0.501, 0.503, 0.505, 0.507, 0.509, 0.511, 521, 0.55, 0.6, 0.7, 0.8, 0.9\}$ and $\beta \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$.

VCL favored $\sigma_n^2 = 0.1$, and we set this value for all other methods. VBS (K=1, VGS) selected $\beta = 0.5$ and $p = 0.513$. VBS (K=3) selected $p = 0.507$ and $\beta = 0.6$. VBS (K=6) selected $p = 0.505$ and $\beta = 0.7$. In generating the plots on the example trajectory, we used $p = 0.507$ with varying β and varying beam size K.

E.7.3 Bayesian Deep Learning Experiments

We performed the Bayesian Deep Learning experiments on a server with Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz and Nvidia TITAN RTX GPUs. Regarding the running time, VCL, and Independent task (Bayes) takes five hours to finish training; Independent task takes three hours; VGS takes two GPUs and five hours; VBS (K=3) takes six GPUs and five hours; VBS (K=6) takes six GPUs and 10 hours. When utilizing multiple GPUs, we implemented task multiprocessing with process-based parallelism.

Datasets with Covariate shifts We used two standard datasets for image classification: CIFAR-10 [Krizhevsky et al., 2009] and SVHN [Netzer et al., 2011]. We adopted the original training set and used the first 5000 images in the original test set as the validation set and

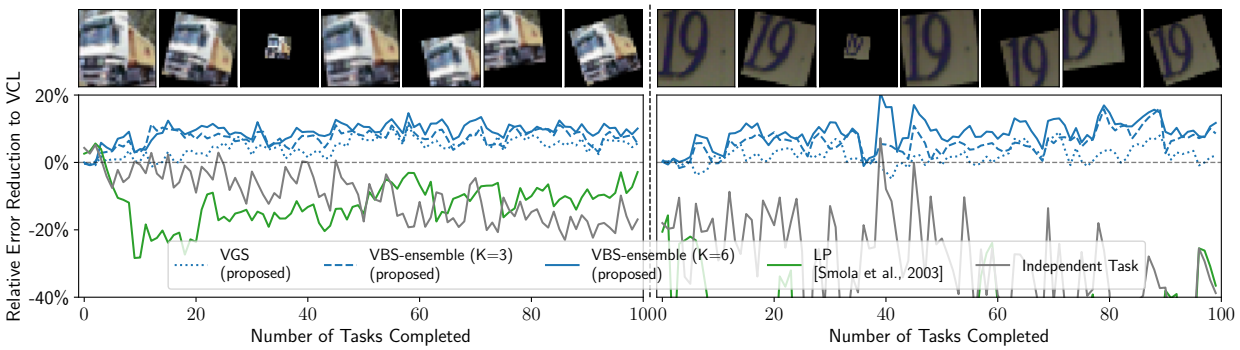


Figure E.5: (**Bottom**) Running test performance of our proposed VBS and VGS algorithms compared to various baselines on transformed CIFAR-10 (left) and SVHN (right). (**Top**) Examples of transformations that we used for introducing covariate shifts.

Table E.1: Convolution Neural Network Architecture

LAYER	FILTER SIZE	FILTERS	STRIDE	ACTIVATION	DROPOUT
CONVOLUTIONAL	3×3	32	1	RELU	
CONVOLUTIONAL	3×3	32	1	RELU	
MAXPOOLING	2×2		2		0.2
CONVOLUTIONAL	3×3	64	1	RELU	
CONVOLUTIONAL	3×3	64	1	RELU	
MAXPOOLING	2×2		2		0.2
FULLYCONNECTED		10		SOFTMAX	

Table E.2: Hyerparameters of Bayesian Deep Learning Models for CIFAR-10

MODEL	LEARNING RATE	BATCH SIZE	NUMBER OF EPOCHS	β	ξ_0 OR λ^{-1}	T
LP	0.001	64	150	N/A	N/A	N/A
BOCD	0.0005	64	150	N/A	0.3	20000
BF	0.0005	64	150	0.9	N/A	20000
VCL	0.0005	64	150	N/A	N/A	N/A
VBS	0.0005	64	150	2/3	0	20000

Table E.3: Hyperparameters of Bayesian Deep Learning Models for SVHN.

MODEL	LEARNING RATE	BATCH SIZE	NUMBER OF EPOCHS	β	ξ_0 OR λ^{-1}	T
LP	0.001	64	150	N/A	N/A	N/A
BOCD	0.00025	64	150	N/A	0.3	20000
BF	0.00025	64	150	0.9	N/A	20000
VCL	0.00025	64	150	N/A	N/A	N/A
VBS	0.00025	64	150	2/3	0	20000

the others as the test set. We further split the training set into batches (or tasks in the continual learning literature) for online learning, each batch consisting of a third of the full data. Each transformation (either rotation, translation, or scaling) is generated from a fixed, predefined distribution (see below for **Transformations**) as covariate shifts. Changes are introduced every three tasks, where the total number of tasks was 100.

Transformations We used Albumentations [Buslaev et al., 2020] to implement the transformations as covariate shifts. As stated in the main paper, the transformation involved rotation, scaling, and translation. Each transformation factor followed a fixed distribution: rotation degree conformed to $\mathcal{N}(0, 10^2)$; scaling limit conformed to $\mathcal{N}(0, 0.3^2)$; and the magnitude of vertical and horizontal translation limit conformed to Beta$(1, 10)$, and the sampled magnitude is then rendered positive or negative with equal probability. The final scaling and translation factor should be the corresponding sampled limit plus 1, respectively.

Architectures and Protocol All Bayesian and non-Bayesian methods use the same neural network architecture. We used a truncated version of the VGG convolutional neural network (in Table E.1) on both datasets. We confirmed that our architecture achieved similar performance on CIFAR10 compared to the results reported by Zenke et al. [2017] and Lopez-Paz and Ranzato [2017] in a similar setting. We implemented the Bayesian models using TensorFlow Probability and the non-Bayesian counterpart (namely Laplace Propagation)

using TensorFlow Keras. Every bias term in all the models were treated deterministically and were not affected by any regularization.

We initialize each algorithm by training the model on the full, untransformed dataset. During every new task, all algorithms are trained until convergence.

Tempered Conditional ELBO In the presence of massive observations and a large neural network, posterior distributions of change variables usually have very low entropy because of the very large magnitude of the difference between conditional ELBOs as in Eq. 6.6. Therefore change variables become over confident about the switch-state decisions. The situation gets even more severe in beam search settings where almost all probability mass is centered around the most likely hypothesis while the other hypotheses get little probability and thereby will not take effect in predictions. A possible solution is to temper the conditional ELBO (or the marginal likelihood) and introduce more uncertainty into the change variables. To this end, we divide the conditional ELBO by the number of observations. It is equivalent to set $T = 20000$ in Eq. 6.6. This practice renders every hypothesis effective in beam search setting.

Hyperparameters, Initialization, and Model Training The hyperparameters used across all of the models for the different datasets are listed in Tables E.2 and E.3. Regarding the model-specific parameters, we set ξ_0 to 0 for both datasets and searched β in the values $\{5/6, 2/3, 1/2, 1/4\}$ on a validation set. We used the first 5000 images in the original test set as the validation set, and the others as the test set. We found that $\beta = 2/3$ performs relatively well for both data sets. Optimization parameters, including learning rate, batch size, and number of epochs, were selected to have the best validation performance of the classifier on one independent task. To estimate the change variable s_t's variational parameter, we approximated the conditional ELBOs 6.7 by averaging 10000 Monte Carlo samples.

As outlined in the main paper, we initialized each algorithm by training the model on the full, untransformed dataset. The model weights used a standard Gaussian distribution as the prior for this meta-initialization step.

When optimizing with variational inference, we initialized $q(\mathbf{z}_t)$ to be a point mass around zero for stability. When performing non-Bayesian optimization, we initialized the weights using Glorot Uniform initializer [Glorot and Bengio, 2010]. All bias terms were initialized to be zero.

We performed both the Bayesian and non-Bayesian optimization using ADAM [Kingma and Ba, 2015]. For additional parameters of the ADAM optimizer, we set $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for both data sets. For the deep Bayesian models specifically, which include VCL and VBS, we used stochastic black box variational inference [Ranganath et al., 2014, Kingma and Welling, 2014, Zhang et al., 2018]. We also used the Flipout estimator [Wen et al., 2018] to reduce variance in the gradient estimator.

Predictive Distributions We evaluated the most likely hypothesis' predictive posterior distribution of the test set by the following approximation:

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathcal{D}_{1:t}, s_{1:t}) \approx \frac{1}{N}\sum_{n=1}^{N} p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{z}_{s_{1:t}}^{(n)})$$

where N is the number of Monte Carlo samples from the variational posterior distribution $q^*(\mathbf{z}_t|s_{1:t})$. In our experiments we found $S = 10$ to be sufficient. We take $\arg\max_{\mathbf{y}_t} p(\mathbf{y}^*|\mathbf{x}^*, , \mathcal{D}_{1:t})$ to be the predicted class.

LP only used the MAP estimation $\mathbf{z}_t^*$ to predict the test set: $p(\mathbf{y}^*|\mathbf{x}^*, \mathcal{D}_{1:t}) \approx p(\mathbf{y}^*|\mathbf{x}^*, , \mathbf{z}_t^*)$.

Standard Deviation in the Main Text Table 6.1 The results in this table were summarized and reported by taking the average over tasks. Each algorithm's confidence, which

is usually evaluated by computing the standard deviation across tasks in stationary environments, now is hard to evaluate due to the non-stationary setup. These temporal image transformations will largely affect the performance, leaving the blindly computed standard deviation meaningless since the standard deviation across all tasks represents both the data transformation variation and the modeling variation. To evaluate the algorithm's confidence, we proposed a three-stage computation. We first segment the obtained performance based on the image transformations (in our case, we separate the performance sequence every three tasks). Then we compute the standard deviation for every performance segment. Finally, we average these standard deviations across segments as the final one to be reported. In this way, we can better account for the data variation in order to isolate the modeling variation.

Running Performance. We also reported the running performance for both our methods and some baselines for each task over time (100 tasks in total) in Fig. E.5. In the bottom panel, to account for varying task difficulties, we show the percentage of the error reduction relative to VCL, a Bayesian online learning baseline. Our proposed approach can achieve 10% error reduction most of the time on both datasets, showing the adaptation advantage of our approach. The effect of beam search is also evident, with larger beam sizes consistently performing better. The top panel shows some examples of the transformations that we used for introducing covariate shifts manually.

E.7.4 Dynamic Word Embeddings Experiments

We performed the Dynamic Word Embeddings experiments on a server with Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz and Nvidia TITAN RTX GPUs. Regarding the running time, for qualitative experiments, Google Books and Congressional Records take eight GPUs and about 24 hours to finish; UN Debates take eight GPUs and about 13 hours to finish. For quantitative experiments, since the vocabulary size and latent dimensions are smaller, each

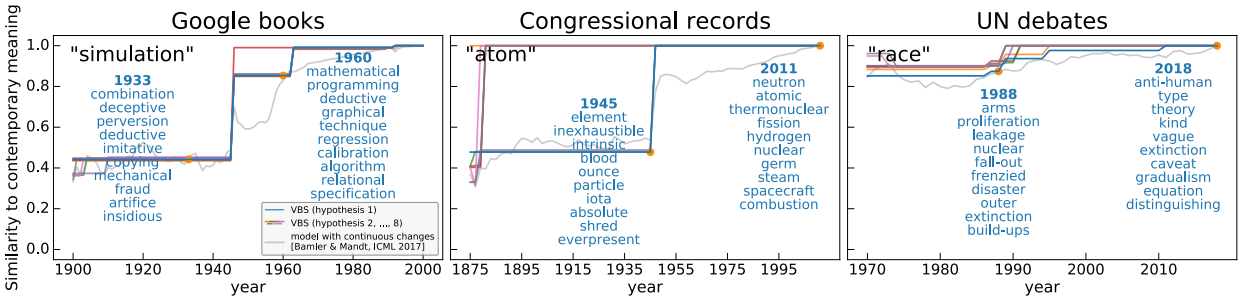


Figure E.6: Dynamic Word Embeddings on Google books, Congressional records, and UN debates, trained with VBS (proposed, colorful) vs. VCL (grey). In contrast to VCL, VBS reveals sparse, time-localized semantic changes (see main text).

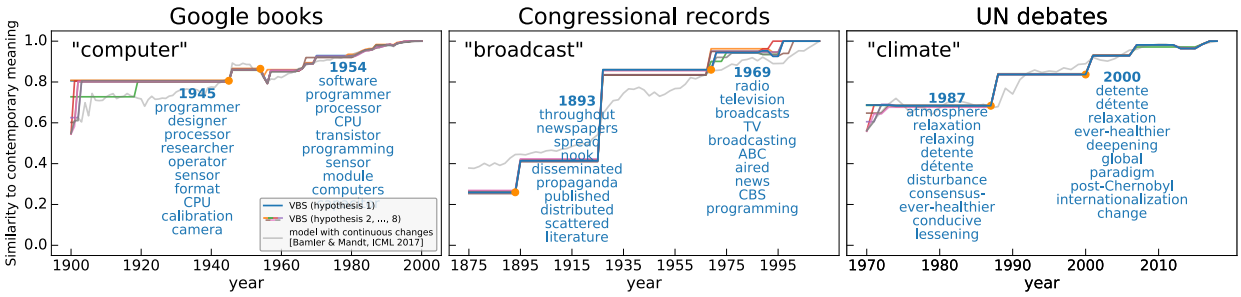


Figure E.7: Additional results of Dynamic Word Embeddings on Google books, Congressional records, and UN debates.

model corresponding to a specific ξ_0 takes eight GPUs and about one hour to finish. When utilizing multiple GPUs, we implemented task multiprocessing with process-based parallelism.

Data and Preprocessing We analyzed three large time-stamped text corpora, all of which are available online. Our first dataset is the Google Books corpus [Michel et al., 2011] consisting of n-grams, which is sufficient for learning word embeddings. We focused on the

Table E.4: Hyperparameters of Dynamic Word Embedding Models

CORPUS	VOCAB	DIMS	β	LEARNING RATE	EPOCHES	ξ_0	BEAM SIZE (K)	T
GOOGLE BOOKS	30000	100	0.5	0.01	5000	-10	8	1
CONGRESSIONAL RECORDS	30000	100	0.5	0.01	5000	-10	8	1
UN DEBATES	30000	20	0.25	0.01	5000	-1	8	1

Table E.5: ξ_0 of Document Dating Tasks

CORPUS	ξ_0
GOOGLE BOOKS	-1000000, -100000, -5120, -1280, -40
CONGRESSIONAL RECORDS	-100000, -1280, -320, -40
UN DEBATES	-128, -64, -32, -4

period from 1900 to 2000. To have an approximately even amount of data per year, we sub-sampled 250M to 300M tokens per year. Second, we used the Congressional Records data set [Gentzkow et al., 2018], which has 13M to 52M tokens per two-year period from 1875 to 2011. Third, we used the UN General Debates corpus [Jankin Mikhaylov et al., 2017], which has about 250k to 450k tokens per year from 1970 to 2018. For all three corpora, the vocabulary size used was 30000 for qualitative results and 10000 for quantitative results. We further randomly split the corpus of every time step into training set (90%) and heldout test set (10%). All datasets, Congressional Records[8], UN General Debates[9], and Google Books[10] can be downloaded online.

We tokenized Congressional Records and UN General Debates with pre-trained Punkt tokenizer in NLTK[11]. We constructed the co-occurence matrices with a moving window of size 10 centered around each word. Google books are already in Ngram format.

Model Assumptions As outlined in the main paper, we analyzed the semantic changes of individual words over time. We augmented the probabilistic models proposed by Bamler and Mandt [2017] with our change point driven informative prior (Eq. 6.3 in the main paper) to encourage temporal sparsity. We pre-trained the *context* word embeddings[12] using the

[8] `https://data.stanford.edu/congress_text`

[9] `https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/0TJX8Y`

[10] `http://storage.googleapis.com/books/ngrams/books/datasetsv2.html`

[11] `https://www.nltk.org/`

[12] We refer readers to [Mikolov et al., 2013, Bamler and Mandt, 2017] for the difference between target and context word embeddings.

whole corpus, and kept them constant when updating the *target* word embeddings. This practice denied possible interference on one target word embedding from the updates of the others. If we did not employ this practice, the spike and slab prior on word i would lead to two branches of the "remaining vocabulary" (embeddings of the remaining words in the vocabulary), conditioned either on the spike prior of word i or on the slab prior. This hypothetical situation gets severe when every word in the vocabulary can take two different priors, thus leading to exponential branching of the sequences of inferred change points. When this interference is allowed, the exponential scaling of hypotheses translates into exponential scaling of possible word embeddings for a single target word, which is not feasible to compute for any meaningful vocabulary sizes and number of time steps. To this end, while using a fixed, pre-trained context word embeddings induces a slight drop of predictive performance, the computational efficiency improves tremendously and the model can actually be learned.

Hyperparameters and Optimization Qualitative results in Figure E.6 in the main paper were generated using the hyperparameters in Table E.4. The initial prior distribution used for all latent embedding dimensions was a standard Gaussian distribution. We also initialized all variational distributions with standard Gaussian distributions. For model-specific hyperparameters β and ξ_0, we first searched the broadening constant β to have the desired jump magnitude observed from the semantic trajectories mainly for medium-frequency words. We then tuned the bias term ξ_0 to have the desired change frequencies in general. We did the searching for the first several time steps. We performed the optimization using black box variational inference and ADAM. For additional parameters of ADAM optimizer, we set $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for all three corpora. In this case, we did **not** temper the conditional ELBO by the number of observations (correspondingly, we set $T = 1$ in Eq. 6.6 in the main paper).

Quantitative results of VGS in Figure 6.2 (c) in the main paper were generated by setting a smaller vocabulary size and embedding dimension, 10,000 and 20, respectively for all three corpora. We used an eight-hypothesis (K=8) VBS to perform the experiments. Other hyperparameters were inherited from the qualitative experiments except ξ_0, whose values used to form the rate-distortion curve can be found in Table E.5. We enhanced beam diversification by dropping the bottom two hypotheses instead of the bottom third hypotheses before ranking. On the other hand, the baseline, "binning", and had closed-form performance if we assume (i) a uniformly distributed year in which a document query is generated, (ii) "binning" perfectly locates the ground truth episode, and (iii) the dating result is uniformed distributed within the ground truth episode. The $L1$ error associated with "binning" with episode length L is $\mathbb{E}_{t\sim\mathcal{U}(1,L),t'\sim\mathcal{U}(1,L)}[|t-t'|] = \frac{L-1}{2}$. By varying L, we get binning's rate-distortion curve in Figure 6.2 (c) in the main paper.

Predictive Distributions In the demonstration of the quantitative results, i.e., the document dating experiments, we predicted the year in which each held-out document's word-word co-occurrence statistics $\mathbf{x}$ have the highest likelihood and measured $L1$ error. To be specific, for a given document in year t, we approximated its likelihood under year t' by evaluating $\frac{1}{|V|}\log p(\mathbf{x}_t|\mathbf{z}^*_{t'})$, where $\mathbf{z}^*_{t'}$ is the mode embedding in year t' and $|V|$ is the vocabulary size. We predicted the year $t^* = \arg\max_{t'} \frac{1}{|V|}\log p(\mathbf{x}_t|\mathbf{z}^*_{t'})$. We then measured the $L1$ error by $\frac{1}{T}\sum_i^T |t_i - t_i^*|$ given T truth-prediction pairs.

Additional Results

Qualitative Results As outlined in the main paper, our qualitative result shows that the information priors encoded with change point detection is more interpretable and results in more meaningful word semantics than the diffusion prior of [Bamler and Mandt, 2017]. Here we provide a more detailed description of the results with more examples. Figure E.6 shows

three selected words ("simulation", "atom", and "race"–one taken from each corpus) and their nearest neighbors in latent space. As time progresses the nearest neighboring words change, reflecting a semantic change of the words. While the horizontal axis shows the year, the vertical axis shows the cosine distance of the word's embedding vector at the given year to its embedding vector in the last available year.

The plot reveals several interpretable semantic changes of each word captured by VBS. For example, as shown by the most likely hypothesis in blue for the Congressional Records data, the term "atom" changes its meaning from "element" to "nuclear" in 1945–the year when two nuclear bombs were detonated. The word "race" changes from the cold-war era "arms"(-race) to its more prevalent meaning after 1991 when the cold war ended. The word "simulation" changes its dominant context from "deception" to "programming" with the advent of computers. The plot also showcases various possible semantic changes of all eight hypotheses, where each hypothesis states various aspects.

Additional qualitative results can be found in Figure E.7. it, again, reveals interpretable semantic changes of each word: the first change of "computer" happens in 1940s–when modern computers appeared; "broadcast" adopts its major change shortly after the first commercial radio stations were established in 1920; "climate" changes its meaning at the time when Intergovernmental Panel on Climate Change (IPCC) was set up, and when it released the assessment reports to address the implications and potential risks of climate changes.

Quantitative Results and Baseline Figure 6.2 (c) in the main paper shows the results on the three corpora data, where we plot the document dating error as a function of allowed changes per year. For fewer allowed semantic changes per year, the dating error goes up. Lower curves are better.

Now we describe how the baseline "Binning" was constructed. We assumed that we had separate word embeddings associated with episodes of L consecutive years. For T years in total, the associated memory requirements would be proportional to $V * T/L$, where V is the vocabulary size. Assuming we could perfectly date the document up to L years results in an average dating error of $\frac{L}{2}$. We then adjusted L to different values and obtained successive points along the "Binning" curve.