6-8-2022 11:00 AM

Solving Challenges in Deep Unsupervised Methods for Anomaly Detection

Vahid Reza Khazaie, *The University of Western Ontario*

Supervisor: Yalda Mohsenzadeh, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Computer Science

Recommended Citation

Khazaie, Vahid Reza, "Solving Challenges in Deep Unsupervised Methods for Anomaly Detection" (2022). *Electronic Thesis and Dissertation Repository*. 8564.
https://ir.lib.uwo.ca/etd/8564

Abstract

Anomaly Detection (AD) is to identify samples that differ from training observations in some way. Those samples that do not follow the distribution of normal data are called outliers or anomalies. In this thesis, we examined two different challenges related to deep learning-based anomaly detection methods. The first challenge is the generalizability to outliers. A wide range of unsupervised anomaly detection methods use deep autoencoders as a foundation. However, a notable limitation of deep autoencoders is that they generalize to outliers and reconstruct them with low error. In order to overcome this issue, we propose an adversarial framework consisting of two competing components, an adversarial distorter, and an autoencoder. During training, the adversarial distorter produces perturbations that are applied to the encoder's latent space to maximize the reconstruction error. The autoencoder attempts to neutralize the effects of these perturbations to minimize the reconstruction error. Another challenge is the high computational cost, complexity, and unstable training procedures of deep anomaly detection methods. Despite being successful at anomaly detection, deep neural networks are difficult to deploy in real-world applications because of this challenge. We overcome this problem by using a simple learning procedure that trains a lightweight convolutional neural network. We propose to solve anomaly detection as a supervised regression problem. We label normal and anomalous data using two separable distributions of continuous values. As a way to compensate for the lack of anomalous samples during training, we use straightforward image augmentation techniques to create a distinct set of anomalous samples. An augmented set has a distribution that is similar to normal data, but deviates slightly from it, while real anomalies should have a further distribution. Consequently, training a regressor on normal and these augmented samples will result in more distinct distributions of labels for normal and real anomalous data points. In several image and video anomaly detection benchmarks, our methods outperform cutting-edge approaches.

Keywords: Anomaly Detection, Deep Learning, Unsupervised Methods

Summary for Lay Audience

The goal of anomaly detection is to recognize samples that differ in some way from the regular observations. Those samples that are out of the distribution of normal data are called anomalies or outliers. Anomaly detection problems have been largely solved well by deep neural networks. However, there are some challenges to developing deep learning-based anomaly detection methods. In this thesis, we have touched upon some critical aspects of anomaly detection methods that have been neglected. The first challenge is generalizability to outliers, while the second challenge is deployability. To address the aforementioned challenges, we propose two different solutions in chapters three and four. The proposed methods achieve state-of-the-art results when applied to anomaly detection tasks.

Co-Authorship Statement

Chapter three of this thesis presents a paper that has been accepted for publication at the 2022 Conference on Computer and Robot Vision (CRV). The paper was written in collaboration with Anthony Wong, John Taylor Jewell, and Yalda Mohsenzadeh. As the first author, I contributed to the ideation of the project, the development of the method, the experiments and analyses, and the writing of the manuscript. Anthony contributed to coding and discussions throughout the project, as well as writing the manuscript. John contributed to ideation and discussions as well. Yalda contributed to the revision of the manuscript. The fourth chapter presents the results of another paper, which is available on the arXiv. This research was conducted collaboratively with Anthony Wong and Yalda Mohsenzadeh. I am the first author and I have implemented the proposed method and contributed to the ideation, conducting of experiments, and writing the manuscript. Anthony contributed to the implementation, the experiments, and writing of the manuscript. Yalda contributed to the revision of the manuscript.

Acknowlegements

I would like to acknowledge the following people who have helped me conduct this research:

I would like to thank the faculty and staff in the Department of Computer Science at Western University, especially the Department Chair, Dr. Michael Domaratzki, the Graduate Chair, Dr. Marc Moreno Maza, and the Graduate Affairs Coordinator, Ms. Janice Wiersma, for their support, assistance, and guidance during this MSc program.

As well, I would like to thank Prof. Michael Bauer, Prof. Charles Ling and Prof. Boyu Wang for reading my thesis and providing me with constructive feedback.

I am thankful for the friendship and support of Anthony, Bardia, Aida, Mohammad, Yassamin, Hessam, John, Masoud, Homa and Rouzbeh.

Lastly, but not least, I would like to thank my family, especially my parents, for their love, support, and encouragement through the years. Without you, none of this would have been possible.

Contents

Abstract i

Summary for Lay Audience ii

Co-Authorship Statement iii

Acknowlegements iv

List of Figures vii

List of Tables ix

1 Introduction and Literature Review 1
- 1.1 Introduction . . . 1
- 1.2 Anomaly Detection . . . 3
- 1.3 Reviews of Previous Anomaly Detection Methods . . . 6
  - 1.3.1 Reconstruction-based . . . 7
  - 1.3.2 Adversarial Training . . . 8
  - 1.3.3 Anomaly Generation . . . 10
  - 1.3.4 Knowledge-Distillation . . . 10
- 1.4 Benchmarks . . . 11
- 1.5 Contributions . . . 12
- 1.6 Conclusion . . . 14

2 Basics of Deep Learning 15
- 2.1 Neural Network Architectures . . . 15
- 2.2 Neural Network Training . . . 17
- 2.3 Data Augmentation . . . 18

3 Adversarially Learned Perturbations of Latent Space 19
- 3.1 Overview . . . 19
- 3.2 Motivation . . . 19
- 3.3 Method . . . 20
  - 3.3.1 Adversarially Learned Perturbations (ALPS) . . . 20
  - 3.3.2 Adversarial Training . . . 22
  - 3.3.3 Anomaly Scores . . . 22
- 3.4 Experiments and Results . . . 23

3.4.1 MNIST and FMNIST 23
3.4.2 UCSD 23
3.4.3 Effectiveness of the Perturbations 23
3.5 Conclusion 25

4 Augment to Detect Anomalies with Continuous Labelling 26
4.1 Overview 26
4.2 Motivation 27
4.2.1 Complexity of Deep Methods for Anomaly Detection 27
4.2.2 Anomaly Generation for Detection 27
4.3 Method 28
4.3.1 Augment to Detect Anomalies with Continuous Labelling (ADACL) . . 28
4.3.2 Regression for Anomaly Detection 28
4.3.3 Continuous Labelling 29
4.3.4 Anomaly Creation with Augmentation 31
4.4 Experiments and Results 32
4.4.1 MNIST, FMNIST and CIFAR-10 32
4.4.2 UCSD 35
4.4.3 BCE vs. MSE 35
4.4.4 Continuous vs. Discrete Labelling 36
4.4.5 Effects of Augmentations 36
4.4.6 Interval Selection 38
4.5 Conclusion 38

5 Conclusion 39
5.1 Conclusion and Discussions 39
5.2 Applications 40
5.3 Limitations 40
5.4 Future Research 40

Bibliography 42

Curriculum Vitae 47

List of Figures

1.1 This figure illustrates the general concept of AD. In this case, the pedestrian is considered normal, while the car is an anomaly. *F* produces a representation of the given data. As can be seen, pedestrians, which are marked by green dots, follow the distribution of the target class. Thus, the car indicated by a red dot is an instance outside the distribution. By measuring *D*, the calculated distance of the car from the normal data shows the deviation. 4

1.2 General demonstration of an autoencoder. As a result, input data is compressed as it passes through the encoder. Then, the output of the encoder will be passed through the decoder and the image will be reconstructed. 7

1.3 The Mask Module learns to cover the important parts of the input image adversarially; it consists of a threshold unit and an autoencoder that build a binary mask from an activation map. Mask Module aims to maximize reconstruction error and Reconstructor aims to minimize it. 9

1.4 Each row represents one class in each dataset. In the top, the left image is the MNIST dataset, the right image is the FMNIST dataset, and the bottom image shows the CIFAR-10 dataset. 11

1.5 These are the four frames from the UCSD anomaly detection dataset. Pedestrians are considered as normal, while vehicles, skateboards, wheelchairs, etc., are considered anomalies. 12

3.1 Overview of ALPS architecture. From the input, the Adversarial Distorter learns how to produce perturbations. The Autoencoder minimizes reconstruction error, while the Adversarial Distorter maximizes it. 21

3.2 A visual demonstration of the performance of our method on MNIST and FMNIST datasets. The left image is the input to the model and the right image is the reconstruction of the model from the perturbed latent representation. 25

3.3 (Left) The first five rows are normal and the last five rows are abnormal patches from UCSD frames. These patches are used as the input to the model. (Right) The first five rows are reconstruction of normal and the last five rows are reconstruction of abnormal patches from UCSD frames. 25

4.1 Sigmoid function and the first derivative of the sigmoid function. Rate of change of sigmoid decreases to 0 as predictions approach its target. 29

4.2 The first derivatives of MSE and BCE. Notice that the gradients of MSE become larger than BCE after a certain point. With mean square error, X represents the difference between two terms, while with binary cross-entropy, it represents a probability. 30

4.3 Overview of ADACL architecture. Normal examples and created anomalies are assigned continuous value intervals and fed into the CNN. This regression model outputs a continuous value between 0 and 1 as an anomaly score. At test time, the model is evaluated with real anomalies. 30
4.4 Various Augmentations applied to create anomalies. The first, second, and third rows contain normal and created anomalies with different augmentations of images from UCSD [8], MNIST [25] and CIFAR-10 [24], respectively. . . . 31
4.5 Predictions of the model on normal, augmented and anomalous samples from UCSD, MNIST, and CIFAR-10. 33
4.6 3D visualization of learned representation of class 1 and 8 of the MNIST dataset. As shown, there are separable and distinct distributions of normal and anomalous embeddings. 33
4.7 Average test AUROC taken over 10 training runs on CIFAR-10. MSE achieves higher AUROC in a shorter number of epochs. 36
4.8 Variance of Average Validation AUROC over all classes in each epoch. The averages are taken over 10 training runs of CIFAR-10. It can be observed that continuous labelling consistently produces lower variance in higher epochs. Thus, a stopping criteria based on the validation set yields more stable test AUROC when continuous labels are used. 37
4.9 Variance in anomaly detection AUROC when using continuous labelling versus discrete labelling for all classes taken over 10 training runs of CIFAR-10. More often than not, using the same stopping criteria with continuous labels produces lower variance in test AUROC. 37
4.10 The effects of the augmentations like Cut-paste, puzzling, one and three times rotation and various mix-ups on AUROC. The most effective augmentation was Cut-paste, however using all augmentations in the process of creating outliers yields the highest performance in anomaly detection. 38

List of Tables

3.1 AUROC in % for anomaly detection on MNIST [25] and FMNIST [61] datasets. 24
3.2 Frame-level AUCROC and EER comparison % on UCSD dataset with state-of-the-art methods. 24
3.3 Comparison of various autoencoders for anomaly detection on MNIST 24

4.1 AUROC in % for anomaly detection on MNIST [25], Fashion-MNIST [61] and CIFAR-10 [24] datasets. 34
4.2 Frame-level AUCROC and EER comparison % on UCSD dataset with state-of-the-art methods. 34
4.3 Comparison of AUROC in % for anomaly detection on MNIST [25] and CIFAR-10 [24] datasets with knowledge distilation methods. 35
4.4 Experiments on different intervals 36

Chapter 1

Introduction and Literature Review

1.1 Introduction

The first chapter of the thesis examines recent advances and challenges in the field, as well as the gaps that encourage this thesis. We also provide an explanation of how the proposed methods addressed the gaps. Afterward, we discuss previous works and standard benchmarks. In the next chapter, we briefly explained some background information. In chapter 3 and 4, we propose two different methods to solve two important challenges in deep learning-based anomaly detection methods. Lastly, we discussed different aspects of these methods and future research directions in chapter 5.

In Anomaly Detection (AD), we look for samples that differ from the training data. Outliers or anomalies are samples which do not conform to the distribution of normal data. The outliers in real-world anomaly detection problems are either absent or poorly defined, or have a limited number of instances. Compared with traditional solutions, deep learning-based methods achieve excellent results in terms of performance. However, developing these approaches presents some challenges. In this thesis, we discussed a few critical aspects of anomaly detection methods that often are missed.

In the rest of this section, we will briefly introduce some related topics, including supervised learning, classification and regression problems, and associated metrics, as well as unsupervised and semi-supervised learning. Our goal is to provide background information on relevant topics to anomaly detection. Later on, we will look at different categories of anomaly detection methods, which vary according to the availability of labels, resulting in either supervised, semi-supervised, or unsupervised learning schemes. In addition, since we need to measure the performance of the anomaly detection methods, we will describe the related metrics we used in chapters 3 and 4 of this thesis, such as AUROC, EER, MSE, etc.

In supervised learning, the objective is to create a mapping function that transforms an input into an output given a set of samples of data. In this learning scheme, every sample contains a set of input objects as well as the desired output value. Classification and regression are two categories of supervised learning algorithms. In classification, data is divided into classes. The concept of classification begins with predicting what the target class, label, or category of a given data point is. As a result, classification models estimate the mapping function from input variables to discrete output variables. Additionally, there are several types of classification tasks, such as binary classification, multiclass classification, imbalanced classification, etc. In contrast, regression is used to approximate a mapping function from input variables to a continuous output variable. As discussed above, regression is a different kind of analysis from classification, which is a way to predict a category or a class label. For instance, predicting stock market prices is a regression problem, but predicting the trend of the market is a classification problem. The regression problem involves predicting a quantity. It is often referred to as multivariate regression when the problem has more than one input variable. Basically, anomaly detection is a classification problem where we are trying to differentiate between anomalous and normal data.

Those that have two classification labels are considered binary classifications. It is typical in anomaly detection tasks to involve a class representing the normal condition (negative class) and a class representing the abnormal condition (positive class). As an example, ”benign” is the state of normal, and ”malignant” is the state of abnormality in tumor detection. Typically, it is assigned a class label of 0 for the normal state, and a class label of 1 for the abnormal state. Often, binary classification tasks are represented with a model that predicts Bernoulli probability distributions for each sample. Bernoulli distribution describes the probability that an observation will result in either a 0 or 1 as its outcome. The model predicts the likelihood of an example belonging to the abnormal class.

Classification with more than two categories are multi-class classification tasks. A few examples are movie genre classification, animal species classification, etc. In contrast to binary classification, multi-class classification omits the concept of normal and abnormal outcomes. Rather, examples are classified as belonging to one of several classes. Those tasks that involve imbalanced classification have unequal distributions of samples per class. An imbalanced classification task is a binary classification that is composed of a vast proportion of examples in the training dataset belonging to a normal class and a slight minority belonging to an abnormal class. Detecting anomalies, outliers, novelty, etc., are examples of this type of classification. As a result of the ambiguity of the abnormal class, solving anomaly detection in a supervised manner is not always feasible.

Different metrics can be used to evaluate the performance of various models in a classification task. These classification metrics include accuracy, precision, recall, F1-score and area under the ROC curve. In classification, accuracy is the number of correct predictions out of all predictions. Usually, classification problems are evaluated based on this metric. This method would only be applicable when each class has almost the same number of observations (which often isn’t the case in the real-world situation) and when the prediction errors are of equal importance. Due to class imbalances, accuracy is usually not a good metric in anomaly detection. Quantifying the number of correct positive predictions is called precision and recall is a metric that calculates the number of correct positive predictions over all possible positive predictions.

The Receiver Operating Characteristic (ROC) curve shows the performance of a classification model by plotting True Positive Rate (TPR) vs. False Positive Rate (FPR) for all classification thresholds. False Positive (FP) means predicting an event that did not happen, and True Negative (TN) is one where the model correctly predicts the negative outcome. Basically, TPR means recall, and FPR means dividing FP by the addition of FP and TN. When the threshold is lowered, more items will be classified as positive, resulting in both higher false positives and true positives. The Area Under the ROC Curve (AUROC or AUC in general) is calculated by measuring the entire two-dimensional area under the entire ROC curve. AUC determines the overall performance across all possible classification thresholds. This measures the quality of the model's predictions regardless of the threshold used for classification. In this thesis, as a result of the imbalance in the anomaly detection datasets and to follow the protocols of other methods, we calculate AUROC as a metric to measure the performance of our methods. In some anomaly detection benchmarks, Equal Error Rate (EER) is also used to report the performance. In a ROC curve, the EER is the point where the false acceptance rate equals the false rejection rate. As a rule, the lower the equal error rate value, the higher the accuracy of the biometric system.

In regression problems, MSE is an important loss function. In a dataset, the MSE is calculated as the squared difference between predicted and actual values. By squaring these two values, the sign is removed, resulting in a positive error value. In addition, squaring has the effect of magnifying or inflating large errors. This means that the larger the difference between the predicted and actual values, the larger the squared positive error. As a result, MSE as a loss function penalizes models more for larger errors. The RMSE can be thought of as an extension of the mean squared error. RMSE is calculated from the square root of the error, which means the units are the same as those of the actual target value. In contrast, MAE is linear and intuitive. MSE and RMSE punish larger errors more than smaller errors, thereby increasing the mean error score. MAE does not assign various weights to different types of errors; instead, scores increase linearly as errors increase. MAE scores are calculated by averaging the absolute error values. In chapter 4 of this thesis, we used MSE to measure the loss of the model during the training.

A semi-supervised learning method is a type of machine learning. The term refers to a learning problem involving a small portion of labeled samples and a vast amount of unlabeled samples from which a model should learn and make predictions. It is a middle ground between unsupervised and supervised learning. In addition to unlabeled data, the algorithm is provided with some supervision for a small subset of the data. In situations where labeling samples is challenging or expensive, semi-supervised learning algorithms are applied. Semi-supervised algorithms can perform better than supervised algorithms fitted only to labeled training examples. We can leverage this training scheme to identify anomalous samples in some anomaly detection problems by creating a small set of labelled samples.

Unsupervised learning involves discovering patterns from unlabeled data by means of algorithms. As a result of estimation, the machine must build a compact internal representation of the underlying complex structure of the data. While supervised learning relies on the supervision by an expert to do the process, unsupervised methods involve self-organization in which patterns are captured as probability density functions or a combination of statistical parameter selections. There are several primary tasks that unsupervised learning models are utilized for, such as clustering, dimensionality reduction, autoencoding, etc. Due to the lack of anomalous samples, most anomaly detection problems are unsupervised. Throughout chapter 3, we focus on autoencoders as our main research method.

1.2 Anomaly Detection

This section discusses the anomaly detection problem and how it is formulated as well as the challenges in the field. Detecting unusual or rare samples and events in the training data is what is known as Anomaly Detection (AD). This actually refers to the process of discovering unseen concepts. AD is generally associated with a huge number of data instances that follow the normal class distribution. A sample that is similar to the training data is called an inlier. In the alternative, if the sample does not match the distribution defined in the training examples, it is considered an outlier. Therefore, abnormalities may arise from any unknown distribution, leading to a very complex learning process. In order to compensate for the absence of outlier samples, anomaly detection methods seek to model the distribution of inlier data [67]. During the testing stage, the deviation from such a model is what reveals whether this instance is anomalous. The overall concept of AD is shown in Figure 1.1. The distance between given instances and the distribution of normal data is computed by D, and the feature extractor F maps the raw data to a set of discriminative features.

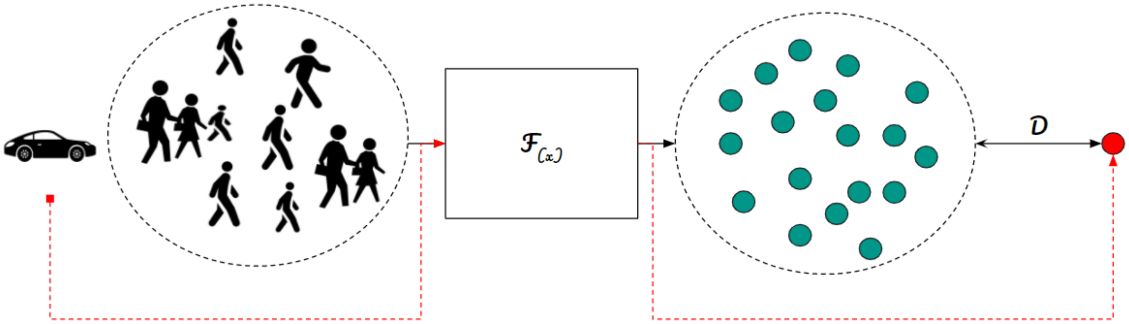


Figure 1.1: This figure illustrates the general concept of AD. In this case, the pedestrian is considered normal, while the car is an anomaly. F produces a representation of the given data. As can be seen, pedestrians, which are marked by green dots, follow the distribution of the target class. Thus, the car indicated by a red dot is an instance outside the distribution. By measuring D, the calculated distance of the car from the normal data shows the deviation.

AD tasks may face varying difficulties depending on the type of data involved. Generalizability to outliers is a weakness of some of the AD algorithms. The detection of abnormal events is considered more critical in most AD applications than the recognition of normal data. In surveillance systems, for instance, ignoring just one anomalous behavior completely compromises the system's reliability and safety. Moreover, some of the AD methods tend to be computationally expensive [13] [36] [43] [45]. A vast majority of deep learning techniques devoted to anomaly detection tasks performed exceedingly well and surpassed the state-of-the-art [42] [46] [43] [45] [21]. However, most of these approaches are too complex to be applied in real-world applications appropriately. These shortcomings confirm that AD tasks are faced with numerous challenges that need to be addressed effectively. Studying AD methods from different perspectives like generalizability, complexity, etc. is an important part of improving the current state-of-the-art.

Anomaly Detection can be considered as special case of Zero-Shot Learning (ZSL) and Open-Set Recognition (OSR). In the Zero-Shot Learning problem, a learner observes samples from non-observed classes at the testing stage and predicts which category they belong to. ZSL methods primarily try to combine knowledge of the seen and unseen classes through some types of auxiliary information. ZSL encodes observable characteristics that distinguish objects. In spite of not having access to the unseen categories during training, it is able to build models by using knowledge from previously seen categories. In Open-Set Recognition, only part of the data is known at training time, and during testing, unknown classes can be evaluated. To accomplish this, the classifiers need to not only classify the seen classes accurately but also deal with the unknown classes effectively. In OSR, algorithms are developed to distinguish between known and unknown data.

For each seen and unseen class, ZSL is usually provided with an additional set of attributes. In contrast, OSR is an open environment setting without information on unseen classes. The objective is to generate a classification model based on observed classes that excludes samples of unseen classes as outliers. In the literature, Novelty Detection (ND) and Anomaly Detection (AD) are used interchangeably. Novelty detection is a special form of open-set recognition in which only one class out of *N* classes is available during the training.

In computer vision, anomaly detection has widespread applications from finding biomedical markers [51] to monitoring video surveillance footage [31] [54] and defect detection [48]. Commonly, images or video frames are unlabeled with the assumption that most of them conform to the distribution of normal data. AD entails determining whether a sample of data follows target distribution or if there is an abnormality. As a next step, we will discuss the rationale behind the methods and explain the different categories of AD techniques.

Explicit learning and utilizing distance metrics are challenging due to the high dimensionality and diversity of data instances in visual domain. The process of learning a discriminative representation from raw data, as well as applying a machine-learning approach in order to learn the distribution of normal data, is required to introduce an effective AD solution. In each of these steps, numerous viewpoints are investigated. This leads to a wide range of proposed algorithms.

As mentioned earlier, the widely used approaches for AD have taken advantage of learning the shared concepts of normal samples. Modeling was initially based on fitting a predefined distribution, such as a Gaussian distribution [44]. The high dimensionality of samples and accordingly the complexity of normal data distribution, however, has led to the use of Deep Neural Networks (DNN) to learn implicitly both the data distribution and detection metric.

The proposed methods for anomaly detection can be classified into three categories, depending on the definition of anomaly and the number of normal, anomalous, and unlabeled samples available in the training set: (1) supervised (2) semi-supervised, and (3) unsupervised. Compared to supervised techniques, unsupervised techniques are more applicable to realistic situations. The following provide a comprehensive explanation of the three major categories determined based on the availability of data labels.

Supervised: Depending on the application, abnormalities can have different explanations, such as intrusion in computer networks [22]. Therefore, to build an efficient binary classifier, normal and anomalous samples should be collected and analyzed in a supervised setting. Although supervised approaches produce highly accurate results, they are not sufficiently generalizable and are not optimal due to the class imbalance in anomaly detection applications. Regardless of the definition used for abnormal events, class imbalances remain a concern. A class imbalance problem occurs when class distributions are highly imbalanced. For infrequent classes, many classification algorithms have low predictive accuracy. This range of solutions is applicable to only a small number of real-world problems.

Semi-Supervised: AD tasks contain a large number of unlabeled data since collecting anomalous instances is a time-consuming and expensive process. In addition, anomalies tend to be very diverse and uncommon. In industrial defect detection, for example, there are many unlabeled samples and few labeled samples. Also, [41, 29] proposes learning a model from lots of unlabeled samples and a few irregular and normal samples, as a means to make use of the abnormal and normal concepts. In general, the notion of having access to the full spectrum of anomalous and normal events for AD application is not attainable and would be costly.

Unsupervised: In an unsupervised setting, the algorithm is trained on unlabeled data. Detection of outliers in this scheme is based exclusively on the internal properties of data examples. The only assumption that can be made in these approaches is that abnormal events are rarely observed. As anomalies in data only occur in rare instances, specifying real abnormalities can be very complex and time-consuming, especially for applications that lack a precise definition of what constitutes an anomalous event. Also, even if anomalies can be accessed, a lack of sufficient data is problematic. The unsupervised methods are essentially One-Class Classification (OCC) tasks aimed at solving AD problems in a more general way.

In a binary classification problem with samples from the positive and negative classes, a machine learning algorithm attempts to differentiate between the two categories. By using this model at test time, unlabeled instances from both groups can be precisely classified. On the other hand, if there is a class imbalance, the percentage of cases in one class will be much higher than the number of samples in the other class. As a result, a typical classifier will tend to favor the negative class since it has a larger number of instances. With traditional binary classifiers, it is very challenging to classify the positive class accurately when the imbalance between classes is severe. To solve such a problem, One-Class Classification (OCC) concepts can be used.

In one-class classification, instances of only one class are examined and analyzed in order to solve the classification problem, and that class is usually of interest in the problem. In an OCC setting, labeled samples of the positive class are either not accessible or are not sufficient to train a supervised classifier. One-class classification appears to be primarily used for outlier detection and novelty detection. In terms of concepts and applications, outlier detection and novelty detection differ slightly. In novelty detection, the anomalies are detected in the test dataset while there are no anomalous data points in the training dataset. Outlier detection occurs when training data includes both normal and abnormal data points. The goal is to determine how to distinguish between them. Afterward, the learned model is applied to the test dataset, which may contain both normal and anomalous samples.

1.3 Reviews of Previous Anomaly Detection Methods

In this section, we review existing works and the literature of anomaly detection. Novelty, outliers, and anomalies are usually identified using one-class classification. When this type of problem is encountered, a model attempts to capture the distribution of inliers and then detect unknown outliers and novel concepts. Traditionally, anomaly detection methods used a one-class SVM [52, 17] and Principal Component Analysis (PCA) [6, 19] to determine which subspace best represents the distribution of normal samples. To formulate the distribution of normal data, unsupervised clustering techniques have also been used, such as k-means [67] and Gaussian Mixture Models [62]. These methods, however, usually fail to deal with high-dimensional visual data. As a result, in the following, we focus on the state-of-the-art deep learning-based anomaly detection methods.

1.3.1 Reconstruction-based

Autoencoders are unsupervised learning techniques that use neural networks to learn representations. With this approach, the goal is to impose a bottleneck in the network in order to force a compressed representation of the original input. The compression and subsequent reconstruction would be very challenging if the input features were independent of one another. Conversely, if a pattern can be found in the data, it can be learned and used to force the input through the network's bottleneck. During training, the network will minimize the reconstruction error, which measures the difference between its original input and its subsequent reconstruction. A key characteristic of our network design is the bottleneck; without it, our network could easily simply learn to memorize input values by passing them along through the network. Figure 1.2 shows the architecture of an autoencoder.

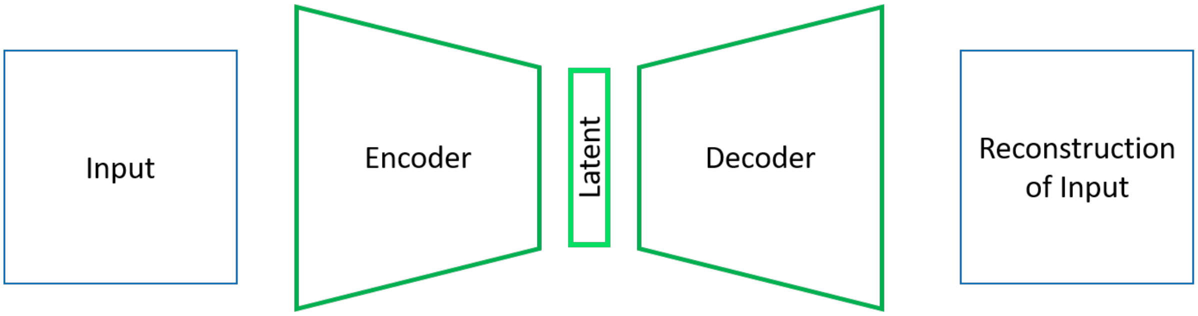


Figure 1.2: General demonstration of an autoencoder. As a result, input data is compressed as it passes through the encoder. Then, the output of the encoder will be passed through the decoder and the image will be reconstructed.

There is a variation on the autoencoder known as a denoising autoencoder [57] that prevents the network from learning the identity function by adding noise to the input. Additionally, if the latent space of the autoencoder is too big, it means that the output will be the same as the input, so the learning of representation or the reduction of dimensionality will not be meaningful in these situations. Denoising autoencoders have been developed to combat this problem by adding noise to the input values during autoencoding. Context Autoencoders (CAEs) are capable of reconstructing the content of randomly masked regions of an input image and can be used for general feature learning. Their representations describe the semantics of the underlying training distribution [68].

A number of previous studies have shown that the reconstruction error of an Autoencoder (AE) can function as a good indicator of whether a sample falls within the distribution defined in the training examples [60]. For this reason, Autoencoders (AE) are often used to detect anomalies in images. A Denoising Autoencoder (DAE) can be used to achieve this goal due to the robust representations and the interpretable anomaly score they provide [2]. It has been shown that context autoencoders (CAEs) have been more successful in detecting anomalies owing to their ability to represent the semantics of the underlying distribution [68]. In general, one of the biggest drawbacks of AEs is that they often generalize well to reconstruct outliers well, which sometimes results in poor performance.

Another type of method, such as reconstruction-based approaches, can also take advantage of learning representation from the input itself. The hypothesis is usually based on the fact that outlier samples cannot be accurately reconstructed by a model that only learns the distribution of inlier samples. Using sparse representations for distinguishing between inlier and outlier samples, Cong et al. [10] proposed a video anomaly and outlier detection model. As outlined in [63, 42], test samples are reconstructed using inlier representations, and reconstruction error is used as a novelty detection metric. The reconstruction loss scales between 0 and 1, so a normal sample will have a score close to 0 and an anomaly will have a score close to 1. In fact, the majority of deep learning-based models that use encoder-decoder architectures [47, 65, 66, 69, 9] have also used this score to detect anomalies. It is important to note that although these encoder-decoder based methods are effective, they are limited due to their insufficient representation of latent space for anomaly detection.

Another category of methods for anomaly detection leverages memory module. In a memory network, the inference capabilities of the neural network model are applied to a memory component, which can be read from and written to. Often, neural networks do not have a mechanism to retain long-term information. It is most likely due to the fact that their existing memory component is insufficient and poorly organized to allow them to properly recall past facts. In order to resolve this issue, memory networks are proposed as a solution. Each memory network is comprised of a memory, which is an array of objects, and four potentially learned components, such as input feature map, generalization, output feature map, and response. Suppose you have an input x, convert it to an internal representation of features, update memory items with the new input, and compute output features based on the new input and memory. Finally, decode output features to give the final outcome. This process is applied at both the training and testing stages.

The application of memory-augmented networks has been widespread [16, 49, 59]. Graves et al. [16] adopted content-based attention to increase the capacity of neural networks. Santoro et al. [49] used a memory network to record information stably. In some works [23, 27] the external memory has also been used for multi-modal data generation. Gong et al. proposed a deep autoencoder augmented with a memory module (MemAE) [13] to encode the input to a latent space with the encoder. The resulting latent vector is used as a query to retrieve the most relevant memory item for reconstruction with the decoder. Also, in [34], they introduced a memory module with items that capture prototypical models of inlier class with a new update system.

1.3.2 Adversarial Training

Some methods for anomaly detection uses adversarial learning framework. A generative adversarial network is a model that consists of two neural networks - a Discriminator network (D) and a Generator network (G) - that work together to learn complex distributions. Basically, what a generator network endeavors to do is to generate realistic data from a random noise vector z. The discriminator's aim is to differentiate real data from fake data generated by the generator. As a result, it is necessary for the discriminator to output a high score for the real data, and a low score for the fake data, in order to achieve the previous objective. The network design is one of the most important aspects of GANs. For example in visual domain, the generator network should take random noise and map it to images in such a way that the discriminator can't determine which images come from the training dataset and which ones are fake. DCGAN (Deep Convolutional Generative Adversarial Networks) is a type of GAN that utilizes convolutional neural networks as their Generator and Discriminator. This architecture achieves superior performance on a variety of computer vision tasks.

In [51], AnoGAN is a generative adversarial network (GAN) that makes use of deep convolutional generative adversarial networks (GAN) to learn a manifold of biomedical images and to calculate an abnormality score based on the mapping from image space to a random distribution of biomedical images. Basically, what the function is supposed to do is to calculate the corresponding latent mapping z for particular queries. In order to obtain the latent mapping corresponding to the given pattern, they begin by selecting a random vector, z, and update its value by backpropagating the loss function. A shortcoming of this method is that calculating the predicted latent vector of each test image with the use of optimization can take up a considerable amount of time. Hence, the f-AnoGAN [50] is proposed for fast unsupervised anomaly detection with the goal of eliminating this disadvantage. A learned encoder is used in f-AnoGAN instead of the previous optimization procedure which requires many iterations. It dramatically accelerates the mapping of images into latent space, thereby making it possible to map more images in a shorter period of time.

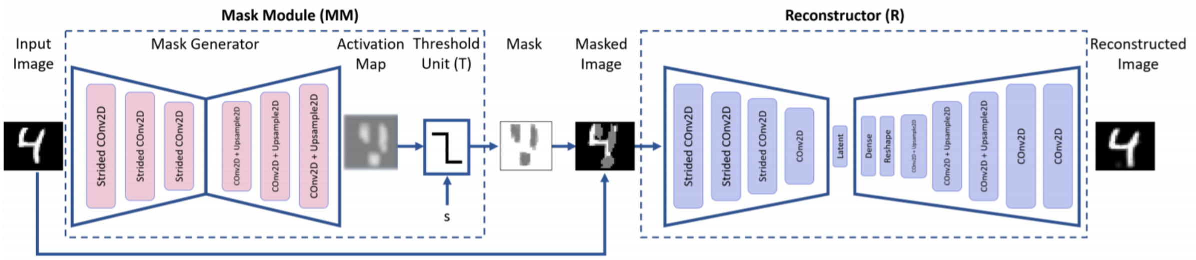


Figure 1.3: The Mask Module learns to cover the important parts of the input image adversarially; it consists of a threshold unit and an autoencoder that build a binary mask from an activation map. Mask Module aims to maximize reconstruction error and Reconstructor aims to minimize it.

In [45], Sabokrou et al. proposed a one-class classification framework which is composed of a Reconstructor (R) and a Discriminator (D). A denoising autoencoder performs the function of R, while a detector performs the function of D. These two networks are learned adversarially in an end-to-end perspective. R reconstructs the noisy input image X and generates X' to deceive D so that it contemplates the reconstructed sample as the original data. On the other hand, D tries to detect reconstructed samples from the original data by accessing the training set and learning their distribution. Both modules form a one-class classifier for end-to-end novelty detection by playing a minimax game.

As an extension of the ALOCC [45], Zaheer et al. [64] further improved the adversarial one-class classifier training setup by modifying the role of the discriminator to differentiate between good and poor quality reconstructions. The network pre-training followed the same methodology by utilizing the same architecture. Then, they provided two different kinds of fake images to fine-tune the networks. They introduced bad quality examples x_{low} and pseudo anomaly images x_{pseudo} to generate bad quality examples x_{low}. To this end, an older state of the generator network is used. The pseudo anomaly image is created by interpolating between two images of poor quality in the image space. They found that the second phase of training resulted in more distinct scores for inliers and outliers.

Perera et al. used auto-encoder networks to enforce the normal samples to be distributed uniformly across the latent space. This model which also benefits from the GAN setup is called OCGAN [36]. It consists of a denoising auto-encoder which maps a noisy input to the latent space and learns an inverse mapping back to the image space, a latent discriminator which learns to distinguish between latent space of real images and noisy samples, a visual discriminator which learns to discriminate between fake images and real images, and a classifier which learns to recognize real images from fake images. They firstly trained the classifier for a given input x while the parameters for other networks are fixed. For the next step, two discriminator losses L_{latent} and L_{visual} are computed and their weights are updated accordingly. Then, negative mining is conducted in the latent space by utilizing the classifier's guidance. Particularly, the latent sample is adjusted using backpropagation such that it may deceive the learned classifier. Finally, L_{latent}, L_{visual} and L_{mse} are used to update auto-encoder weights.

As mentioned earlier, CAEs try to reconstruct a randomly masked image. Although context masking can enhance the performance of AEs in anomaly detection by learning the semantic context but most of the time, random masking fails to cover important parts of the input, leading to learn suboptimal representation. To overcome this issue, OLED [21] is introduced. In this method, they utilized adversarial setup to train two autoencoders. The first network learns to mask the input intelligently and other one is a context autoencoder that inpaints the input. The former is called $MaskModule$ and it generates masks to increase the reconstruction loss when applied to the input and the latter is called *Reconstructor* that tries to minimize that loss.

1.3.3 Anomaly Generation

In some real-world cases, anomaly detection methods based on learning the distribution of inliers cannot be applied in practice. Anomalies can be generated alongside normal data during training to turn the problem into a supervised classification task. A GAN can be used to create anomalous data, which turns anomaly detection into a binary classification problem. According to [37], they trained a Wasserstein GAN on normal data and used the generator before convergence had been reached. In this way, the generated data deviates from inliers in a controlled manner. Even though they offer new avenues for detecting anomalies, training a neural network to generate outliers is computationally expensive.

1.3.4 Knowledge-Distillation

In addition, there are methods known as knowledge-distillation methods that attempt to utilize the knowledge of a larger network. [48] and [12] have attempted to benefit from deep pre-trained networks by distilling the knowledge where a small student model learns from a large teacher model. In [48], they utilized a VGG16 [53] to calulate a multi-layer loss from different activations for training the student network to calculate the anomaly score. They also incorporate interpretability algorithms in their framework to localize anomalous regions and perform anomaly segmentation. Even though these methods could perform anomaly detection with high performance, they are benefiting from the knowledge achieved by training on millions of labeled images which is not always helpful like other modalities of data.

1.4 Benchmarks

This section provides a detailed explanation of four datasets that are benchmarks in anomaly detection. In this thesis, we chose MNIST [25], FMNIST [61], CIFAR-10 [24] and UCSD [61] for anomaly detection. These benchmark datasets are widely used in the anomaly detection literature. In the following, we provide descriptions of each dataset as well as the protocols for evaluation.

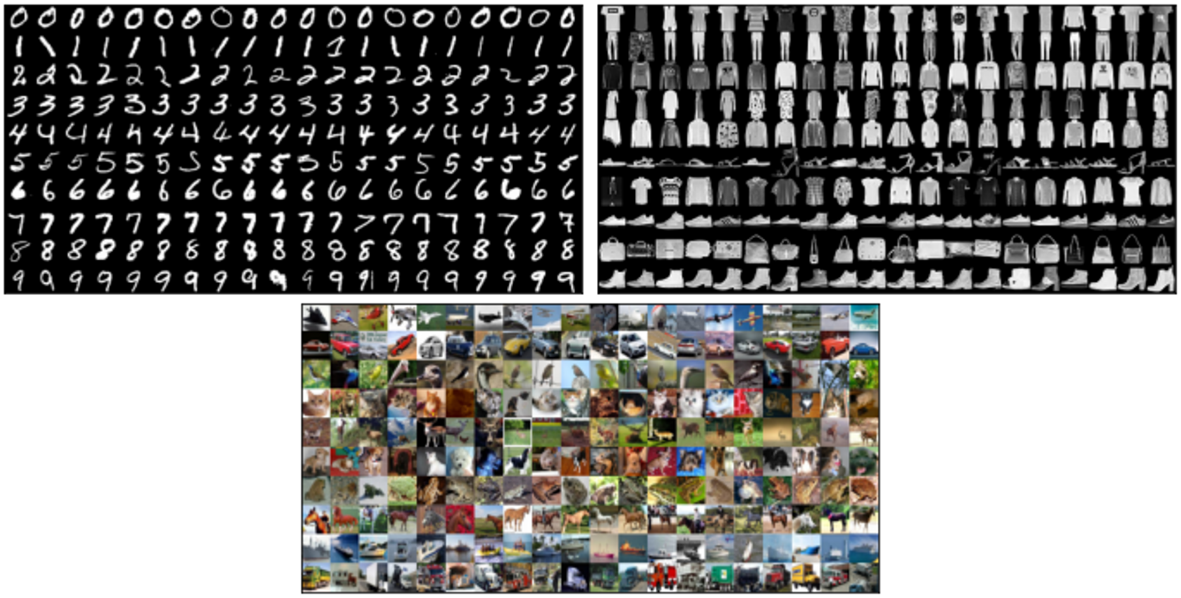


Figure 1.4: Each row represents one class in each dataset. In the top, the left image is the MNIST dataset, the right image is the FMNIST dataset, and the bottom image shows the CIFAR-10 dataset.

MNIST: MNIST is a dataset of handwritten digits with 60,000 grayscale images with a resolution of 28 by 28. This is one of the most popular benchmark datasets in anomaly detection.

FMNIST: A similar dataset, FMNIST contains 60,000 images of 28 by 28 grayscale fashion accessories but since there is a substantial amount of intra-class variation, it is more challenging for anomaly detection than MNIST.

CIFAR-10: This dataset consists of 10 classes of 32 × 32 RGB images of natural objects. With high intra-class variance, CIFAR-10 is a more challenging benchmark for anomaly detection.

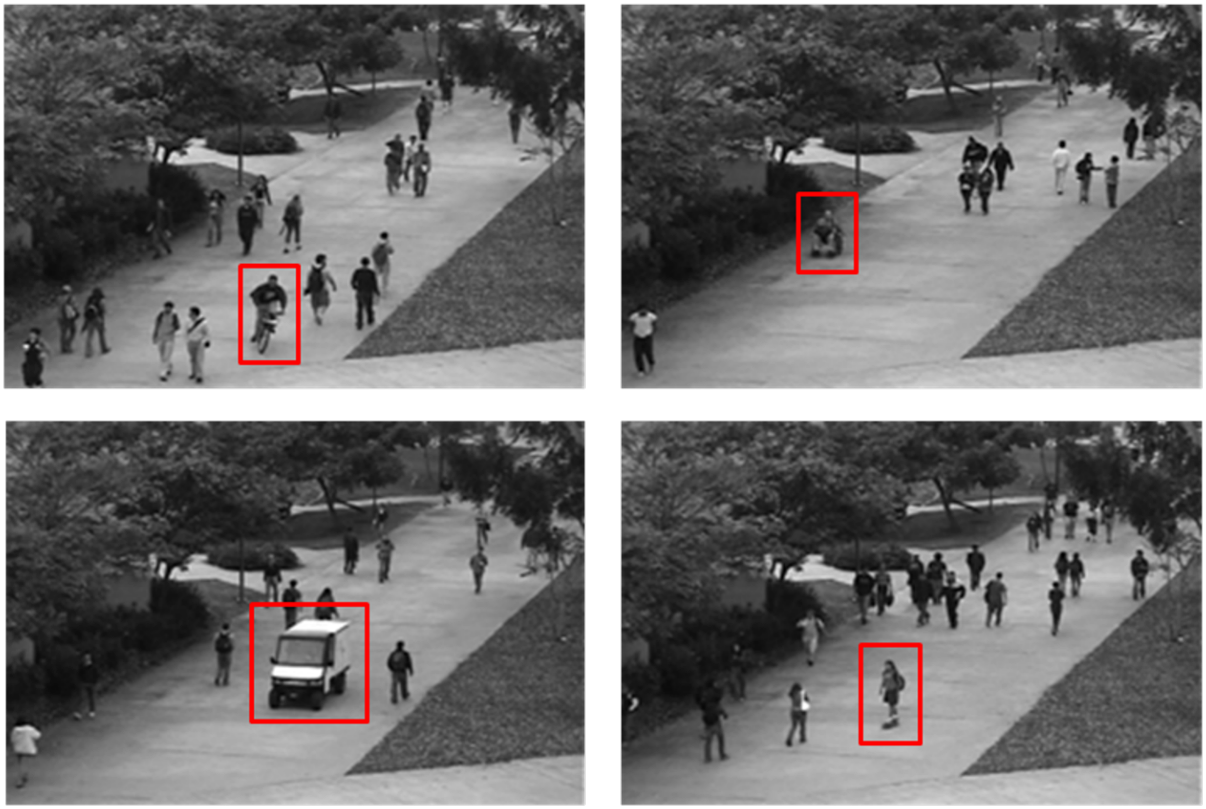


Figure 1.5: These are the four frames from the UCSD anomaly detection dataset. Pedestrians are considered as normal, while vehicles, skateboards, wheelchairs, etc., are considered anomalies.

For these three datasets, the protocol we follow is to designate one class as normal data and other classes as outliers. As a measure of anomaly detection performance, we calculate the Area Under the Curve (AUROC) for each class and report the average for all classes. The examples of these datasets are shown in Figure 1.4.

UCSD: UCSD video dataset contains outdoor scenes with pedestrians, cars, skateboards, wheelchairs, and bicycles. A normal frame is defined as only containing pedestrians, whereas an anomalous frame contains other objects. This dataset has two subsets named Ped1 and Ped2. Ped1 contains 34 training videos and 36 testing videos, while Ped2 contains 2,550 frames in 16 training videos and 2,010 frames in 12 testing videos, all of which have a resolution of 240 by 360 pixels. This dataset is evaluated using a patch-based protocol where each frame is divided into 30 by 30 sections. In training, only pedestrians are included in the patches, yet, the model was evaluated on patches that contained pedestrians or other objects at the test time. The examples of this dataset are shown in Figure 1.5

1.5 Contributions

As mentioned in Section 1.3.1, there are some anomaly detection algorithms that generalize well to outliers. In anomaly detection, generalizability to outliers can be problematic. In most

AD applications, abnormal events need to be detected. The reliability of a system could be compromised by ignoring just one anomalous sample, for instance, in medical image analysis. In many unsupervised anomaly detection methods, deep autoencoders serve as the basis. However, one notable shortcoming of deep autoencoders is that they cannot provide adequate representations for anomaly detection by reconstructing outliers with low error rates. We propose a method in chapter 3 of this thesis to solve this issue. This method leverages an adversarial framework that consists of two competing components, an Adversarial Distorter and an Autoencoder. An Adversarial Distorter generates effective perturbations from a convolutional encoder and an autoencoder reconstructs the images from the perturbed latent space using a convolutional architecture. During training, the networks are trained with opposing goals: the adversarial distorter produces perturbations to be added to the encoder's latent feature space to maximize reconstruction error, and the autoencoder tries to neutralize these perturbations in order to minimize this loss. By applying perturbations to the feature space, the proposed anomaly detection method learns semantically richer representations.

In real-world anomaly detection problems, outliers often don't exist, are poorly defined, or have a very small number of occurrences. Deep neural networks have been effective in solving anomaly detection problems. However, they often have many parameters, so training them can be challenging and costly. As explained in Sections 1.3.2 and 1.3.4 of this chapter, anomaly detection methods based on deep learning suffer from high computational costs, complexity, and unstable training procedures, making them difficult to use in real-world applications. Our solution, in chapter four of this thesis, utilizes a simple learning procedure to train a lightweight convolutional neural network, thus achieving state-of-the-art performance in anomaly detection. This chapter proposes the solution of anomaly detection using supervised regression. We use two separate distributions of continuous values to label normal and anomalous data. To compensate for the lack of anomalous samples during training time, we create a separate set of samples as anomalies using simple image augmentation techniques. As expected, the distribution of the augmented data is similar to that of the normal data, but it deviates slightly from it, while real anomalies are expected to deviate even further. Thus, training a regression model on normal and these augmented samples will result in more distinguishable distributions between samples of normal and anomalous data points.

The results of several experiments conducted on image and video datasets demonstrate the superiority of the proposed methods over the state-of-the-art approach to anomaly detection. The last chapter also discussed the limitations and applications of each of these methods. The contributions of this thesis can be summarized as follows:

- We propose Adversarially Learned Perturbations of Latent Space (ALPS) that leads to richer representations by reconstructing outliers with higher error to improve the performance of autoencoders.

- We propose a simple yet effective approach called Augment to Detect Anomalies with Continuous Labelling (ADACL), which includes a lightweight CNN trained with regression, anomaly creation with augmentations, and continuous labelling to improve performance stability in anomaly detection.

1.6 Conclusion

When applied to anomaly detection tasks, deep neural networks can achieve state-of-the-art performance. Previous studies have shown that the reconstruction error of Autoencoders (AEs) can be used to determine whether or not a sample follows the distribution specified in the training examples [60]. Therefore, AEs are commonly used for anomaly detection. However, the main disadvantage of AEs is that they generalize too well to outliers and therefore learn insufficient representations for anomaly detection. Furthermore, most of these methods suffer from expensive computations, high complexity, and training instability. This thesis proposes two different solutions to rectify these problems. We present Adversarially Learned Perturbations (ALPS) for the training of AEs that overcomes the generalizability problem. ALPS allows AEs to learn more effective representations to detect anomalies. To learn these representations, adversarially generated perturbations will be added to the latent space of the autoencoder. In addition, we propose an effective and simple methodology (ADACL) for anomaly detection in order to alleviate the complexity issue. We convert the problem into a supervised regression task by creating anomalies using data augmentations and training a lightweight convolutional neural network over continuous labels. Our results on several image and video anomaly detection benchmarks demonstrate our superiority to cutting-edge methods.

Chapter 2

Basics of Deep Learning

2.1 Neural Network Architectures

The focus of this section is on neural networks, their architectures, and main components. A Neural Network (NN) is a combination of linear functions that attempt to uncover the connections between a set of data and identify patterns through a procedure that emulates the way the human brain works. The term neural network alludes to a group of neurons, either artificial or biological in nature. With neural networks, the input can change, and the network can generate the best possible result without having to redesign the output criteria. Over the past few years, deep neural networks have become increasingly popular. They are used in a variety of applications within the computer vision domain, such as image classification, object detection, etc. The term deep networks refers to neural networks with multiple layers of processing. Neural networks come in a variety of different forms, such as Perceptron, Multi-layer Perceptron (MLP), Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN).

The perceptron is a supervised learning algorithm for linear binary classification problems with a single layer. They are neural network units that recognize features in input data by carrying out specific computations. Basically, it is a function that maps its input, which is multiplied by the weight coefficients and returns an output value. The activation function checks whether the weighting function's output is greater than zero. It outputs a signal if the sum of the inputs reaches a threshold. In accordance with the perceptron learning rule, the algorithm would automatically learn the appropriate weights. Perceptrons, however, can only learn linearly separable patterns.

In Artificial Neural Networks (ANNs), Multi-layer Perceptrons (MLPs) are a class of feedforward neural networks. The term can sometimes be used to refer to feedforward ANNs, however, it is more often used to refer to networks composed of multiple layers of perceptrons. The general rule of thumb is that MLPs should consist of at least three layers of nodes: the input layer,the hidden layer, and the output layer. When only one hidden layer is present in a multi-layer perceptron, it is commonly referred to as a vanilla neural network. A neuron, except for the input nodes, uses a nonlinear activation function. The adjustment of the weights is performed by the backpropagation optimization technique during the training process. The main difference between an MLP and a linear perceptron is in its multi-layer structure as well as its non-linear activation. This structure enables neural networks to be universal function approximators.

The activation function in a neural network specifies the output of a neuron given an input. An example of this is the linear perceptron. However, only nonlinear activation functions are capable of solving complex tasks using only a few neurons, and they are considered nonlinearities. Activation functions can have different types, such as the sigmoid, relu, tanh, etc.

Convolution is considered to be one of the most important topics in the field of image processing. It is an operation that is used to merge two arrays or signals by multiplying them together. There is no restriction as to how big the arrays can be, the only requirement is that these two arrays must be of the same dimension. A kernel is a smaller part of the convolution operation that is used to slide over the larger part. With kernels, the objective is to extract valuable information from the input. The kernel can also be described as a mask or a convolutional matrix. A kernel can be used to achieve many different effects. For example, a kernel can blur an image, sharpen the image, modify the contrast of the image, etc. The first step in convolving the mask over an image is to place the mask center on each element of the image. The next steps are to multiply all the elements of the image, add them all together, and then paste the result on the element where the mask center is placed. In the next step, we take another patch from the input and fill in new pixel values. This creates a new image that has the same characteristics as the original image but in fewer dimensions.

Pooling operation is carried out by sliding a filter over each dimension of the input and summarizing the features that lie within the scope of the filter. Summarizing can be achieved by using maximum or average operations. In this way, the feature maps are reduced in size by discarding some elements in the region of the filter. Thus, there will be fewer parameters to learn and the number of computations will be reduced when using in the neural networks. In the neural networks, the operation layer creates a summarised set of features from which further operations are performed. After carrying out the above operation, the model becomes more robust to changes in the positions of the features in the input image.

In some cases, it can be problematic to train deep neural networks with many layers since the networks can be extremely sensitive to initial random weights and settings of the learning algorithm. In fact, a possible explanation for this difficulty could be that the updated weights following each mini-batch of input data may affect the distribution of inputs for higher layers of the network. Learning algorithms are therefore constantly seeking to capture the meaningful relations and features in the input. Technically, this shift in the distribution of input to layers within the network is called the internal covariate shift. In Batch Normalization (BN), the inputs to each layer of a deep neural network are standardized for each mini-batch. As a result, the learning process is stabilized, thereby dramatically reducing the number of training epochs necessary to train deep networks.

Convolutional Neural Networks (CNN) are different from typical neural networks by their excellent performance with the image input. A convolutional, a pooling, and a fully-connected layer (FC) are the three types of layers in this type of neural network. During training, the CNN becomes more complex with each layer, allowing it to identify more features in the image. The earliest layers focus on simple features, such as colors and edges. The image data is processed by the CNN through successive layers, and as it advances, it detects larger parts or more details of the object until it recognizes the target.

In a CNN, a convolutional layer forms the core building block of computation. Suppose that the input is a 3-dimensional color image made up of a matrix of pixels. The input will have height, width, and depth, which correspond to the RGB color space in an image. Additionally, we have a feature detector, also known as a kernel or filter, that verifies if the feature is present by moving across the receptive fields of the image. The feature detector corresponds to an array of weights that describe a specific portion of the image and is also employed to specify the size of the receptive field. The kernel is then applied to an area of the image, and a dot product between the input pixels and the filter is calculated. This dot product is then fed into an output array. Afterwards, the kernel shifts by a stride, repeating the process until the entire image has been scanned. This final output of dot products is called a feature or activation map.

Each value in feature maps does not need to correspond to every pixel in the input image. The filter only needs to be connected to the receptive field. Convolutional layers can also be described as local connectivity since they do not require the output array to map directly to each input value. Also, since the feature detector's weights remain constant while it moves across the image, the parameters in convolutional layers are shared. During training, weight values are adjusted through backpropagation and gradient descent. Before the CNN can be trained, some hyperparameters that affect the output volume need to be set, such as the number of filters, stride, padding. Input image depth is influenced by the number of filters and stride is the number of pixels the kernel moves over the input matrix to produce the output image. Padding is usually used when filters don't fit the input image depth. In the convolution step, usually a Rectified Linear Unit (ReLU) is applied to the feature map, adding nonlinearity to the model.

In the process of pooling layers, dimensionality is reduced, which decreases the number of parameters in the input. In the same way as the convolution layer, the pooling operation sweeps a filter across the input, but this filter does not have any weights. In contrast, an aggregate function populates an output array with values from the receptive field, which can be either a maximum or average operation. The pooling layer does lose a great deal of information, but it also benefits CNN in a number of ways. As a consequence, they decrease complexity and increase efficiency and robustness. Fully-connected layers can be used to learn non-linear combinations of the high-level features represented by the convolutional layer. In this layer, a possibly non-linear function is being learned.

2.2 Neural Network Training

The purpose of this section is to explain various loss functions and how they are used to optimize neural network weights. Loss functions quantify a difference between the expected outcome and the output of a model. In order to update the weights, we can calculate the gradients from the loss function. Loss functions based on cross-entropy are commonly used in classification problems. Cross-entropy is used to measure the difference between two probability distributions. In regression problems, mean squared error is used where the expected and predicted outcomes are real numbers. Across all samples, it is the average of the squared difference between the expected and predicted value.

The amount of information needed to transmit a randomly selected event from a probability distribution is called entropy. A distribution where events have equal probability has a larger entropy while a skewed distribution (imbalanced distribution) has low entropy. When comparing two distributions, the cross-entropy calculates how many bits are required to represent or transmit an average event. For example, if a probability distribution P is the target or underlying distribution, and Q is an approximation of P, the cross-entropy of Q from P is the number of additional bits needed to represent an event using Q. Binary cross-entropy compares each prediction to the actual output, which is either a zero or a one. Based on the distance from the expected value, it calculates the score that penalizes the probabilities. The score refers to how far the probabilities are from the actual values. To put it another way, Binary Cross Entropy is the negative average of correct predicted probabilities.

Backpropagation is an algorithm for optimizing the weights of neural networks using gradient descent. Given a neural network and a loss function, the algorithm calculates the gradient of the loss function with respect to the neural network's weights. As the name implies, the gradient is computed backward through the network, beginning with a gradient calculated for the weights in the final layer, and ending with a gradient calculated for weights in the first layer. In this way, the gradient can be computed efficiently at each layer due to the backward flow of error information. With the wide adoption of deep neural networks, backpropagation's popularity has recently risen.

2.3 Data Augmentation

The purpose of this section is to describe the process of data augmentation in the image domain and how it is applied to the training of neural networks. With more data available, deep neural networks often perform better. Augmenting existing training data is a technique for producing new data through artificial means. Using domain-specific techniques, we are able to create new and different training samples based on the training data. In the case of image data augmentation, transformed versions of the original images are created that belong to the same class as the transformed versions in the training dataset. These operations include shifts, flips, zooms, and a number of others from the field of image processing.

In order to select the specific data augmentation technique for a dataset, one must carefully consider the dataset and the knowledge of the problem domain before making a decision. Additionally, it can be beneficial to experiment with different data augmentation techniques both independently and in combination, perhaps using a small sample set, to determine if they lead to a measurable improvement in model performance.

In a convolutional neural network (CNN), features can be learned that are invariant to their location in the image. Furthermore, augmentation can support this transform-invariant learning approach and can further assist the model in learning features that are also invariant to transformations. In general, augmentation is only applied on the training dataset, and not the testing or validation dataset.

Chapter 3

Adversarially Learned Perturbations of Latent Space

3.1 Overview

Deep autoencoders serve as a basis for many unsupervised anomaly detection methods. However, one notable shortcoming of deep autoencoders is that they generalize to outliers by reconstructing them with low error. In anomaly detection, generalizing to outliers means the AD model fails to detect them such that they are misclassified as normal data. Although autoencoders are not trained on outliers, they can still reconstruct them with a low error, which results in reduced anomaly detection performance. In this chapter, we will describe what we propose as a way to decrease the generalizability of autoencoders to outliers. In the following section, we will provide a description of how we addressed this issue. Later, we will explain the architecture of our model, the objective functions used during training, and details about the implementation. We will then discuss a set of experiments that we conducted to evaluate the performance of the model. This chapter will conclude with a discussion of the proposed method and its advantages and disadvantages.

3.2 Motivation

Detecting abnormal samples from a group of normal data is the goal of anomaly detection. Here, our goal for anomaly detection is to find those samples which are different than our normal data. Anomaly detection differs from common supervised classification problems due to either poor sampling or inaccessibility of abnormal data during training. Therefore, one-class classification is an efficient approach for solving this task.

The task of detecting anomalies in images involves identifying whether an image is an inlier or an outlier based on training data that primarily consists of inlier images. As a solution to the lack of outlier samples, one-class classification methods attempt to model the distribution of only the inlier data [67]. A sample that does not match the inlier distribution is considered an outlier. The high dimensionality in which the data points exist makes it difficult to model the distribution of image data with conventional methods [67].

Deep learning has contributed to developing methods that effectively produce representations of high-dimensional data [3]. Of these methods, Autoencoders (AEs) are an unsupervised class of algorithms that are suitable for modeling image data [4]. A standard AE consists of two components: an encoder and a decoder. Encoders learn to map images into a latent space, while decoders learn mappings from the latent space to original images. The model weights are optimised by minimizing the error between the original image (input to the encoder) and its reconstruction (output of the decoder).

In many approaches to one-class classification, AEs serve as a powerful unsupervised way to learn representations for anomaly detection [7]. Prior to detecting abnormal images, the AE is trained on sets of mostly normal images. An anomaly score is calculated using the reconstruction error of a sample. As a result, the reconstruction error is expected to be lower for inliers compared to outliers [60]. However, this assumption is not always true, and the AE can reconstruct images outside the distribution of the training data as well [69, 13]. This is particularly evident when abnormal images share patterns with inliers. In anomaly detection, generalizability to outliers can be problematic.

In more recent methods, the autoencoder's reconstruction task has become more complex, so outliers are not reconstructed well [1, 36, 68]. In the Denoising Autoencoders (DAE), the model learns to remove added noise from an input [58]. DAE has also been demonstrated to provide more robust representations. A specific type of DAE, context autoencoders (CAE) [35], has demonstrated excellent performance in anomaly detection. In contrast to adding noise, random masking is applied to input images, and the reconstruction task involves inpainting the randomly masked region. In this way, random masking implicitly forces CAEs to learn semantic information about the distribution of training data [35]. In some cases, CAEs have difficulties detecting anomalies because of suboptimal representations.

3.3 Method

3.3.1 Adversarially Learned Perturbations (ALPS)

In order to improve anomaly detection using autoencoders (AE), we need to reduce the generalizability of the model to outliers so that they are reconstructed with higher error. As a result, they can be detected as anomalies, which improves the overall performance of anomaly detection. In previous studies, the reconstruction error of AEs have been shown to be a good metric of whether or not a sample follows the distribution specified in the training examples [60]. However, the AE can sometimes generalize to outliers and reconstruct them accurately, which compromises the anomaly detection performance using reconstruction error. To alleviate this issue, we designed an adversarial framework consisting of two competing components, an Adversarial Distorter, and an Autoencoder. The adversarial distorter aims to maximize the encoder's reconstruction error by applying perturbations to the latent space, while the autoencoder attempts to minimize it by neutralizing the effects of these perturbations.

The two modules of our framework, the Autoencoder and the Adversarial Distorter, are trained with two opposing goals. During training, the autoencoder aims to reconstruct perturbed inputs with minimal error, whereas the adversarial distorter aims to increase it by adding perturbations to the latent space. To train the networks, the same input is given to both the autoencoder and adversarial distorter. We add the generated perturbation to the latent space of the encoder. The perturbed latent space is then passed to the decoder for reconstruction. The involvement of perturbations introduces a new task for the autoencoder. In addition to reconstruction, the autoencoder will learn to neutralize the effect of added perturbations during training. Since the autoencoder is optimized only on inliers, it is unable to neutralize the effect perturbations added to the latent space of the outliers. Because of this, the autoencoder reconstructs outliers poorly, which reduces its generalizability to outliers. During this procedure, the perturbations allow the autoencoder to learn different variations in the latent space of inliers. When tested with an outlier, it will modify its latent space to be similar to that of inliers, which results in higher reconstruction error. During test time, we pass the input to both modules.

Contrary to previous AE-based approaches, our framework enables autoencoders to learn semantically richer representations of exclusively normal data by overcoming the added perturbation. An overview of the method is shown in Figure 3.1. As part of the training, only normal data is used and no samples from anomalous data are taken.

The size of the latent space for both the adversarial distorter and encoder should also vary to adequately represent each dataset. To avoid the dominance of the adversarial distorter, we update its weights every couple of epochs. In addition, we created a weighted loss for both modules, giving larger values to the reconstruction loss for the autoencoder due to the high complexity of its task. Our method can be broken down into two different components, which we briefly explain in this section.

Autoencoder: The Autoencoder is a convolutional neural network that consists of an encoder and a decoder. The encoder maps the input to a latent space through four convolutional layers followed by a global average pooling. The decoder maps from the latent space back to the image, beginning with a dense and a reshaping layer followed by six transpose convolutional layers for upsampling. An autoencoder is trained to remove the perturbations produced by the adversarial distorter.

Adversarial Distorter: The Adversarial Distorter is a convolutional encoder that generates perturbations from a given input. The perturbations are used to make reconstruction more difficult for the autoencoder.

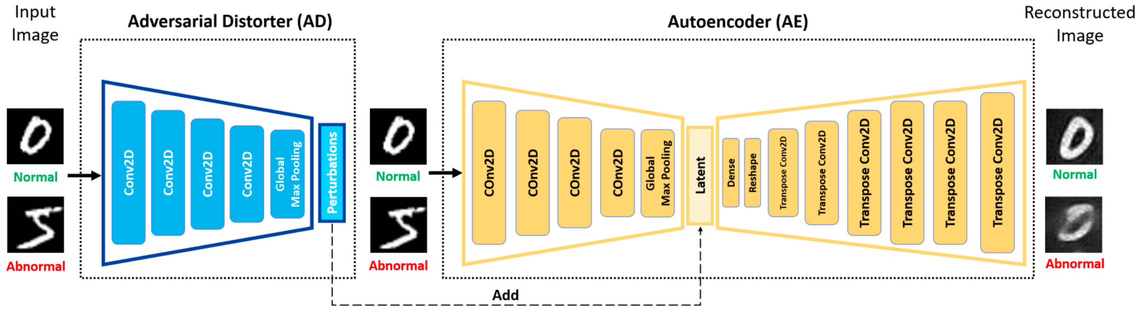


Figure 3.1: Overview of ALPS architecture. From the input, the Adversarial Distorter learns how to produce perturbations. The Autoencoder minimizes reconstruction error, while the Adversarial Distorter maximizes it.

3.3.2 Adversarial Training

The adversarial training mechanism involves two networks competing in a minmax game that progressively enhances their ability to model the underlying distribution of data. This is accomplished by training a generator network G and a discriminator network D in this manner. Taking a noise vector as input, G produces samples that follow the distribution of the training data. As an alternative, D attempts to discriminate between real samples from the training set as well as fake samples generated by G. GANs [14] have the following objective when given an image x sampled from p_{data} and a random latent vector z sampled from p_z:

$$\min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z))] \tag{3.1}$$

$G(z)$ is a sample generated by G with input z and a discriminator's classification scores are $D(x)$ and $D(G(z))$ for a real and generated sample, respectively.

Similarly, the Adversarial Distorter (AD) and Autoencoder (both Encoder and Decoder) are trained in an adversarial optimization scheme. Adversarial Distorter is able to produce a higher reconstruction error from the Autoencoder by creating perturbations. The loss used for training the networks is mean squared difference of the reconstruction of the perturbed image and the input. A perturbed image is one whose latent space has been distorted by the adversarial distorter. Given an inlier image x sampled from p_{data}, objective function of our methods is defined as:

$$\min_{Enc,Dec} \max_{AD} \mathbb{E}_{x\sim p_{\text{data}}(x)} \frac{1}{m} ||x - Dec(Enc(x) + AD(x))||^2 \tag{3.2}$$

where Enc and Dec are components of the Autoencoder, AD denotes the Adversarial Distorter, $Dec(Enc(x)+AD(x))$ is the reconstruction of the perturbed image with the Autoencoder, and m is number of elements in the image.

The loss used for training both the Autoencoder and the Adversarial Distorter is the reconstruction loss. The Autoencoder is trying to minimize this loss while the Adversarial Distorter is maximizing it. Since the autoencoder learned to remove the effect of added perturbations to the latent space only on inliers, it is unable to neutralize that effect on outliers. Therefore, the autoencoder performs poorly with outliers and reconstructs them with higher errors, which decreases its generalizability.

3.3.3 Anomaly Scores

The loss term in ALPS presents an opportunity for three anomaly scores to be defined: the normal reconstruction loss, the perturbed reconstruction loss and the average of them both. Normal reconstruction loss is calculated using mean squared error (MSE) between the input and the reconstruction. In the perturbed reconstruction loss, we calculate MSE while the latent space of the input is perturbed. Both losses are scaled between 0 and 1 to calculate the anomaly scores. The third anomaly score is calculated by averaging between the two losses before scaling. The best score among these three were reported.

3.4 Experiments and Results

3.4.1 MNIST and FMNIST

The following section provides a detailed analysis of the proposed method. The results of ALPS are compared to recent and state-of-the-art methods in the literature on three datasets that are benchmarks in anomaly detection. Throughout all experiments, the method is trained exclusively on inlier samples. Moreover, a validation set containing 150 samples from inliers and 150 samples from outliers from the training set is used to determine the optimal epoch for selecting models.

In this chapter, we chose MNIST [25], FMNIST [61] and UCSD [8] for anomaly detection. These benchmark datasets are widely used in the anomaly detection literature. In the following, we provide descriptions of each dataset as well as the protocols for evaluation.

MNIST is a dataset of handwritten digits with 60,000 grayscale images with a resolution of 28 by 28. This is one of the most popular benchmark datasets in anomaly detection. A similar dataset, FMNIST contains 60,000 images of 28 by 28 grayscale fashion accessories but since there is a substantial amount of intra-class variation, it is more challenging for anomaly detection than MNIST. For these two datasets, following [51][40][52][28][36][1][15], the protocol we follow is to designate one class as normal data and other classes as outliers. As a measure of anomaly detection performance, we calculate the Area Under the Curve (AUROC) for each class and report the average for all classes. Table 3.1 summarizes the AUROC results for each dataset. The results of our study indicate ALPS is superior to the state-of-the-art autoencoder methods for anomaly detection. Figure 3.2 shows the visual performance of model on MNIST and FMNIST datasets, respectively. As shown, the abnormal images are reconstructed similar to normal images after adding perturbations.

3.4.2 UCSD

As part of our method evaluation, we selected the UCSD video dataset. It contains outdoor scenes with pedestrians, cars, skateboards, wheelchairs, and bicycles. A normal frame is defined as only containing pedestrians, whereas an anomalous frame contains other objects. This dataset has two subsets named Ped1 and Ped2. Ped1 contains 34 training videos and 36 testing videos, while Ped2 contains 2,550 frames in 16 training videos and 2,010 frames in 12 testing videos, all of which have a resolution of 240 by 360 pixels. Following [64], this dataset is evaluated using a patch-based protocol where each frame is divided into 30 by 30 sections. In training, only pedestrians are included in the patches, yet, the model was evaluated on patches that contained pedestrians or other objects at the test time. A frame-level AUROC and Equal Error Rate (EER) are calculated in Table 3.2 to report the performance of our method on this dataset. ALPS surpasses state-of-the-art methods for detecting video anomalies on UCSD. The visual performance of the model is depicted in Figure 3.3.

3.4.3 Effectiveness of the Perturbations

Furthermore, we demonstrated the effectiveness of the added perturbations in a complementary experiment. Compared to autoencoders trained with only random noise and a normal autoen-

Table 3.1: AUROC in % for anomaly detection on MNIST [25] and FMNIST [61] datasets.

Dataset	Method	0	1	2	3	4	5	6	7	8	9	Mean
MNIST	AnoGAN[51]	96.6	99.2	85.0	88.7	89.4	88.3	94.7	93.5	84.9	92.4	91.3
	DSVDD[40]	98.0	99.7	91.7	91.9	94.9	88.5	98.3	94.6	93.9	96.5	94.8
	OCSVM[52]	99.5	99.9	92.6	93.6	96.7	95.5	98.7	96.6	90.3	96.2	96.0
	$\text{CapsNet}_{\text{PP}}$ [28]	99.8	99.0	98.4	97.6	93.5	97.0	94.2	98.7	99.3	99.0	97.7
	OCGAN[36]	99.8	99.9	94.2	96.3	97.5	98.0	99.1	98.1	93.9	98.1	97.5
	LSA[1]	99.3	99.9	95.9	96.6	95.6	96.4	99.4	98.0	95.3	98.1	97.5
	Ours (ALPS)	99.68	99.92	94.09	96.19	98.31	97.25	99.64	97.25	95.56	98.65	**97.65**
Dataset	Method	T-shirt	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot	Mean
FMNIST	DAGMM[69]	30.3	31.1	47.5	48.1	49.9	41.3	42.0	37.4	51.8	37.8	41.7
	DSEBM[65]	89.1	56.0	86.1	90.3	88.4	85.9	78.2	98.1	86.5	96.7	85.5
	LSA[1]	91.6	98.3	87.8	92.3	89.7	90.7	84.1	97.7	91.0	98.4	92.2
	DSVDD[40]	98.2	90.3	90.7	94.2	89.4	91.8	83.4	98.8	91.9	99.0	92.8
	OCSVM[52]	91.9	99.0	89.4	94.2	90.7	91.8	83.4	98.8	90.3	98.2	92.8
	Ours (ALPS)	94.42	98.46	89.82	90.65	91.68	90.40	80.43	97.88	97.87	97.88	**92.94**

Table 3.2: Frame-level AUCROC and EER comparison % on UCSD dataset with state-of-the-art methods.

Method	AUCROC (%)	EER (%)
TSC [32]	92.2	-
FRCN action [18]	92.2	-
AbnormalGAN [38]	93.5	13
MemAE [13]	94.1	-
GrowingGas [55]	94.1	-
FFP [30]	95.4	-
ConvAE+UNet [33]	96.2	-
STAN [26]	96.5	-
Object-centric [20]	97.8	-
Ravanbakhsh [39]	-	14
ALOCC [45]	-	13
Deep-cascade [43]	-	9
Old is gold [64]	98.1	7
Ours (ALPS)	**98.16**	**6**

Table 3.3: Comparison of various autoencoders for anomaly detection on MNIST

Method	AUCROC (%)
Vanilla AE	95.42
AE with added random noise	95.95
AE with ALPS	**97.65**

coder, adversarially learned perturbations can significantly improve autoencoders' performance for anomaly detection. Table 3.3 shows the result of this experiment on MNIST dataset.

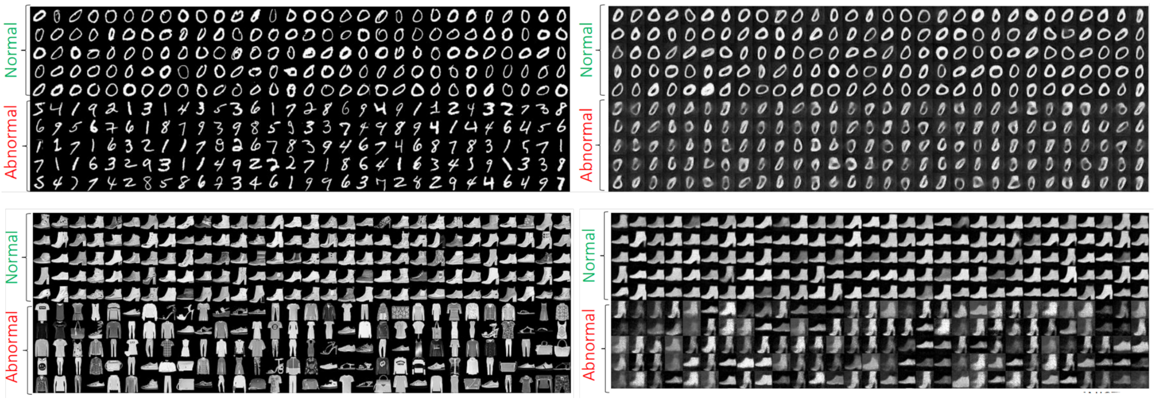


Figure 3.2: A visual demonstration of the performance of our method on MNIST and FMNIST datasets. The left image is the input to the model and the right image is the reconstruction of the model from the perturbed latent representation.

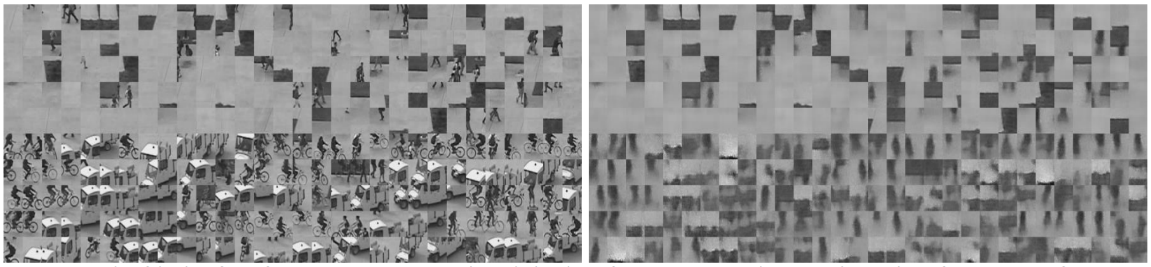


Figure 3.3: (Left) The first five rows are normal and the last five rows are abnormal patches from UCSD frames. These patches are used as the input to the model. (Right) The first five rows are reconstruction of normal and the last five rows are reconstruction of abnormal patches from UCSD frames.

3.5 Conclusion

In this chapter, we presented an adversarial framework for detecting anomalies in both images and videos. In particular, our method includes a convolutional encoder-decoder (Autoencoder) that tries to reconstruct perturbed images and an encoder (Adversarial Distorter) that attempts to generate effective perturbations from input data. The Adversarial Distorter will increase the reconstruction loss by perturbing the latent space of the input, while the Autoencoder attempts to minimize it. Adding perturbations to the latent space of autoencoders decreases the generalization to outliers and improves anomaly detection performance at test time. The results demonstrate that ALPS outperforms recent state-of-the-art models for identifying anomalies.

Chapter 4

Augment to Detect Anomalies with Continuous Labelling

4.1 Overview

Anomaly detection is the problem of identifying abnormal samples among a group of normal data. This is a deviation from many machine learning problems because the set of abnormal data is either poorly sampled or unavailable during training. Recently, anomaly detection draws huge attention and provides many applications in the field of computer vision like marker discovery in biomedical data [51] and crime detection in surveillance videos [31]. Tackling these problems involves modelling the distribution of normal visual samples in a way that anomalies are identified at test time.

Deep neural networks have become a popular choice to reach state-of-the-art performance in anomaly detection. Despite their good performance, these models suffer from high computational cost, complexity, and training instability, making them difficult to use in practice. To overcome these limitations, we propose training a relatively shallow CNN with significantly fewer parameters with continuous labelling which yields state-of-the-art performance on anomaly detection in less training time. Specifically, we approach anomaly detection as a supervised regression problem, where the model's objective is to map normal and created anomalous data to highly separable distributions. In continuous labelling, we designate two continuous intervals for normal and anomalous data to draw labels. Instead of using 0 and 1 to represent normal and anomalous classes, we used a continuous value in this range. In normal samples, the labels are continuous values close to 0 (for example 0.2), while in anomalous samples, they are continuous values near 1 (for example 0.8). For each sample, the value of the label is drawn from the designated interval using a uniform distribution (for example range of 0 to 0.3 for normal samples and range of 0.7 to 1 for anomaly samples).

Due to the unavailability of anomalies, we apply simple data augmentation techniques on normal data to create distinct anomalies. With the new set of anomalous data, we can treat anomaly detection as a supervised learning problem. Since there are now two classes, it is intuitive to treat this as a binary classification problem. However, we show that using regression instead of classification improves anomaly detection performance. Furthermore, we introduce continuous labelling as a favorable means of performance stability. The distribution of the augmented set is similar but slightly deviated from the normal data, whereas real anomalies are expected to have an even further distribution. Therefore, training a regressor on these augmented samples will result in more separable distributions of labels for normal and real anomalous data points.

We evaluated our proposed method, Augment to Detect Anomalies with Continuous Labelling (ADACL), on various benchmark datasets for anomaly detection. ADACL outperforms most state-of-the-art methods using significantly fewer parameters. We also provide a thorough study on loss functions, the choice of labels and the effects of the different augmentations. In this chapter, our contributions are the following:

- We propose a novel method of anomaly detection which includes a lightweight CNN trained with regression, anomaly creation with augmentations and continuous labelling to improve performance stability.
- Our method is simple yet outperforms most state-of-the-art approaches.
- We study the effects of various losses, data augmentations and continuous labelling on anomaly detection performance.

4.2 Motivation

4.2.1 Complexity of Deep Methods for Anomaly Detection

Some of the deep learning-based methods for anomaly detection, which rely on learning the distributions of only inliers to detect anomalies, cannot be applied to real-world problems due to their complexity, training instability, high computational cost, etc.[48][12][13][36][45][64]. These problems make them difficult to deploy in real-world applications. [48] and [12] have attempted to benefit from deep pre-trained networks by distilling the knowledge where a small student model learns from a large teacher model. In [48], they utilized a VGG-16 deep neural network [53] to calculate a multi-level loss from different activations for training the student network to determine the anomaly score. They also incorporate interpretability algorithms in their framework to localize anomalous regions and perform anomaly segmentation. Although knowledge-distillation methods could perform anomaly detection with high performance, they benefit from pre-training on millions of labeled images which is not effective in other modalities of data. Also, in practice, knowledge-distillation methods may not be suitable due to computationally expensive inference. As our proposed method does not use pre-trained networks or contrastive learning paradigms, we consider our work complementary and do not compare it to these approaches.

4.2.2 Anomaly Generation for Detection

Generating anomalies alongside the available normal data build an informative training set for the task of anomaly detection. Employing GANs for generating anomalous data turns the problem of anomaly detection into a binary classification problem. This method can also be used for data augmentation for anomalous data. In [37], they trained a Wasserstein GAN on normal samples and utilized the generator before convergence. In this case, generated irregular data have a controlled deviation from inliers. Although they set a new research direction in anomaly detection, training a network to generate outliers is computationally expensive. Instead of using GANs to generate anomalies, we propose a simpler method that turns the problem into a supervised task. Furthermore, we demonstrated the effectiveness of continuous labelling on various benchmarks.

4.3 Method

4.3.1 Augment to Detect Anomalies with Continuous Labelling (ADACL)

As previously mentioned, deep neural networks often have a large number of parameters, making them difficult and expensive to train. Not only that, most of them suffer from training instability, complexity, and is intractable to deploy in real-world applications. To overcome these issues, we propose Augment to Detect Anomalies with Continuous Labelling (ADACL), where we follow an intuitive and stable training procedure which also exceeds state-of-the-art performance. ADACL is simple to implement and has relatively few parameters, leading to inexpensive training and fast inference time. Therefore, it is more suitable for use in real world scenarios.

4.3.2 Regression for Anomaly Detection

To improve performance in anomaly detection, it is desirable to produce representations of normal and anomalous samples that have distinct distributions. In our method, we redefine anomaly detection as a supervised regression problem. However, the training data consists mainly of normal samples, which makes supervised learning a cumbersome task. To solve this issue, we leverage straightforward data augmentations to create anomalous samples during training. The use of data augmentations is more efficient in terms of computation. Moreover, learning to classify between normal and augmented data is more difficult due to their similarity and when the model learns this task, it can classify real anomalies more accurately since they are more deviated.

We utilize a lightweight convolutional neural network (CNN) to train on the normal and created anomalies. This CNN as a regressor contains four convolutional layers followed by a global max-pooling layer and two fully-connected layers. It receives an image as input and in the last layer, there is one neuron that produces a value ranging from 0 to 1 to represent normal and anomalous data. The network has no activation function on the last layer, and its value is simply clipped between 0 and 1. Even though this can be considered a binary classification problem, we show that regression offers faster convergence and high performance in anomaly detection.

Here we will explain the different configuration options we have for our method. As examples, we explain why we did not use sigmoid in the last layer of the CNN and instead used value clipping as well as why Mean Squared Error (MSE) was chosen over Binary Cross Entropy (BCE) as the loss function. In the case of binary classification, equation 4.3 shows that the rate of change of the sigmoid function is always decreasing as the prediction approaches the ground truth target. Also, as shown in Figure 4.1, the value of gradients are nearing zero in the same manner. Due to the saturation of gradients near the target, updating the weights will be less effective, resulting in slow convergence. Hence, choosing sigmoid as the activation function in the last layer is not appropriate. We instead clipped the values of the last layer of the CNN between 0 and 1. A similar problem exists in binary classification with BCE. Negative log likelihood in BCE also exhibits a decreasing rate of change as the predicted value approaches the ground truth target. Referring to Figure 4.2, MSE provides stronger gradients as the prediction approaches the target, which results in faster convergence. Thus, we selected MSE as the loss function. Moreiver, we perform anomaly detection experiments on ADACL and find not only that MSE converges faster, but also manages to maintain consistently high performance across multiple training runs. Consequently, we transform anomaly detection into a regression problem to reach the optimal solution faster.

$$h(x) = \frac{1}{1+e^{-x}} \tag{4.1}$$

$$h'(x) = h(x)(1 - h(x)) \tag{4.2}$$

$$\begin{aligned} for\ x < 0: \quad & h'(x) - h'(x-1) > 0 \\ for\ x > 0: \quad & h'(x-1) - h'(x) > 0 \end{aligned} \tag{4.3}$$

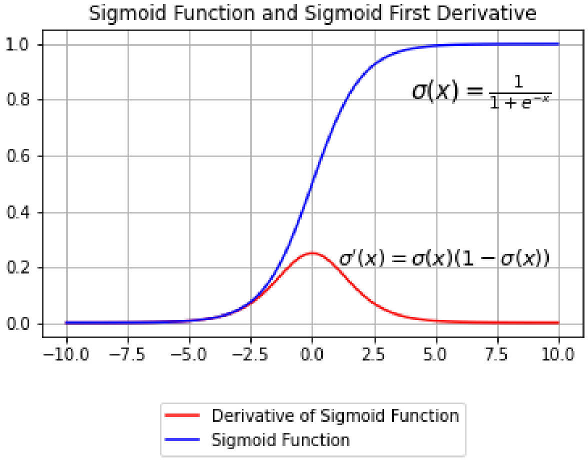


Figure 4.1: Sigmoid function and the first derivative of the sigmoid function. Rate of change of sigmoid decreases to 0 as predictions approach its target.

4.3.3 Continuous Labelling

In the anomaly detection problem, we can label normal and anomalous data as 0 and 1, respectively. We call this Discrete Labelling (DL). However, experimental results show that this leads to high variance in anomaly detection performance. Instead, we use Continuous Labelling (CL), where we designate two continuous intervals corresponding to normal and anomalous

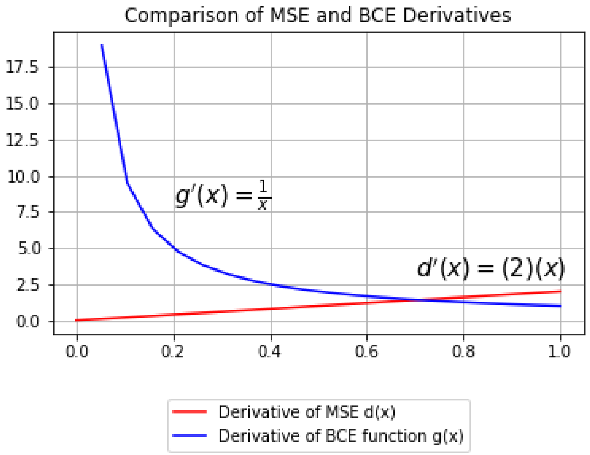


Figure 4.2: The first derivatives of MSE and BCE. Notice that the gradients of MSE become larger than BCE after a certain point. With mean square error, X represents the difference between two terms, while with binary cross-entropy, it represents a probability.

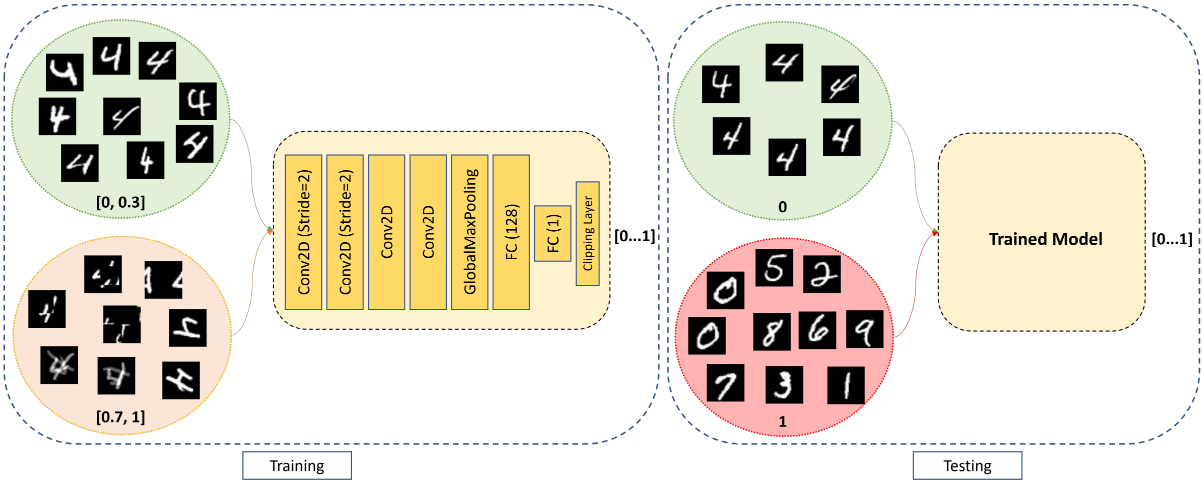


Figure 4.3: Overview of ADACL architecture. Normal examples and created anomalies are assigned continuous value intervals and fed into the CNN. This regression model outputs a continuous value between 0 and 1 as an anomaly score. At test time, the model is evaluated with real anomalies.

data, and sample labels from them using a uniform distribution. The intuition behind continuous labelling is that the expected value of MSE over predictions is lower in comparison to using discrete labelling. Let a discrete label $\in \{0, 1\}$ and a continuous label $\in \{[0, X_L], [X_H, 1]\}$. X_L is the upper bound of the interval of normal class and X_H is the lower bound of the interval of anomaly class. Because we sample from a uniform distribution, $A = \mathbb{E}([0, X_L]) = \frac{X_L}{2}$ and $B = \mathbb{E}([X_H, 1]) = \frac{X_H+1}{2}$. A and B are the expected values of prediction for the normal and anomaly classes, respectively. The MSE function takes two numbers as the input to calculate the loss. As shown in equation 4.4, we let the prediction of our model be 0.5 (highest distance to the lower and upper bounds). The following inequalities show that the value of the MSE loss is always lower when using continuous labelling compared to discrete labelling:

$$\begin{aligned} MSE(0.5, A) &< MSE(0.5, 0)\ if\ A > 0 \\ MSE(0.5, B) &< MSE(0.5, 1)\ if\ 1 > B \end{aligned} \tag{4.4}$$

Therefore:

$$\begin{aligned} &\mathbb{E}(MSE(prediction, CL)) < \\ &\quad \mathbb{E}(MSE(prediction, DL)) \end{aligned} \tag{4.5}$$

According to equation 4.5, if we choose continuous labelling over discrete labelling, the expected value for the loss is lower during training and thus, convergence is slower. Therefore, it should increase training stability. The experimental results in Figures 4.8 and 4.9 supports this hypothesis.

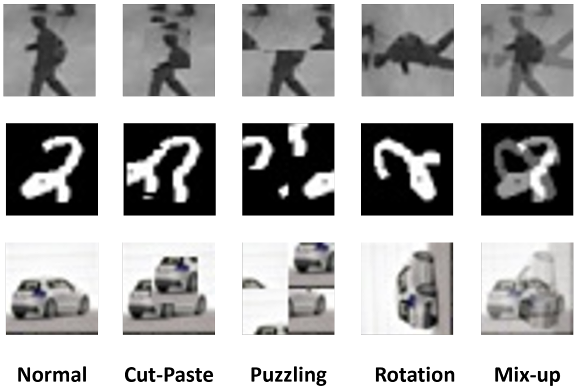


Figure 4.4: Various Augmentations applied to create anomalies. The first, second, and third rows contain normal and created anomalies with different augmentations of images from UCSD [8], MNIST [25] and CIFAR-10 [24], respectively.

4.3.4 Anomaly Creation with Augmentation

Solving anomaly detection as a supervised regression problem requires a dataset containing both normal and anomalous data. To compensate for the unavailability of anomalies, we utilize data augmentation techniques to create them during training. Examples of these augmentations are shown in Figure 4.4. Our goal was to choose image augmentations in a manner that modifies the structure (not color) of the image slightly. The following are descriptions of our proposed augmentations for ADACL:

- ***Cut-Paste:*** Randomly select patch from image and place it in a random location.
- ***Puzzling:*** Take quarters of the image and shuffle them.
- ***Rotation:*** Rotate the image 90 degrees one or three times
- ***Mix-up:*** Add a rotated image to the original one. Prior to adding, the rotated and original image are multiplied by respective coefficients.

To assign training labels to normal and created anomalies, we pick two separate continuous intervals from which we uniformly sample. For example, normal and anomalous labels are in the range of [0, 0.3] and [0.7, 1], respectively.

4.4 Experiments and Results

Our method which is depicted in Figure 4.3, uses a simple CNN with less than 300k parameters. As a regressor, this model outputs a continuous value which is clipped between 0 and 1. It is trained on the inlier samples and created anomalies which are augmented versions of the normal data. We use different variations of the Adam optimizer in conjunction with a cyclic learning rate. Also, we designate a constant number of epochs for training on each dataset. This method uses much fewer epochs than many state-of-the-art methods for all datasets. For example, the model can reach the optimal solution on MNIST [25] dataset in less than 10 epochs. Then, based on the validation set, we use early stopping techniques to terminate training. This validation set consists of 150 randomly selected samples of normal data and the augmented version of them as anomalies. This random selection maintains consistency by using a random seed.

4.4.1 MNIST, FMNIST and CIFAR-10

Following [51][40][52][28][36][1][15], the protocol we follow for these three datasets is to consider one class as normal data and the rest as anomalies. For each inlier in these datasets, we randomly select from the previously explained data augmentations and apply them. Therefore, we have an equal number of samples of normal and generated data for training our model. In the training, we only use normal data and generated anomalies with data augmentation. At the test stage, the model is evaluated with normal data and real anomalies. To measure anomaly detection performance, we calculate Area Under the Curve (AUROC) for each class and report the average of all classes as the final performance. AUROC on these datasets are shown in table 4.1. From our results, ADACL outperforms recent state-of-the-art anomaly detection methods. It is noted that all other methods we compared against also used the same protocol for this experiment. Moreover, Figure 4.5 depicts the model's predictions on normal, augmented and anomalous samples over different datasets. Figure 4.6 shows the 3D distributions of learned representations of normal and anomalous samples on class 1 and 8 of MNIST. This figure shows the separability of learned representations.

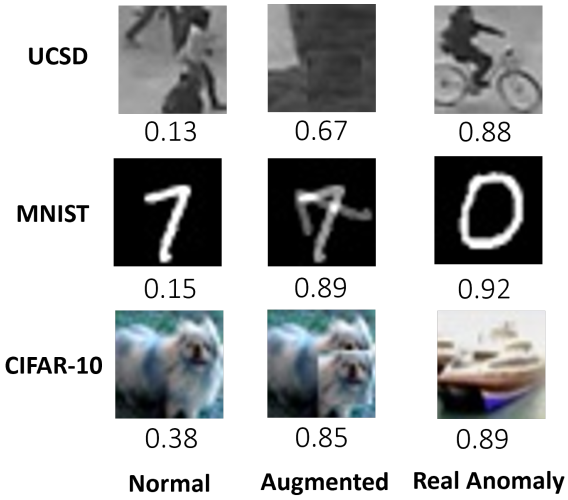
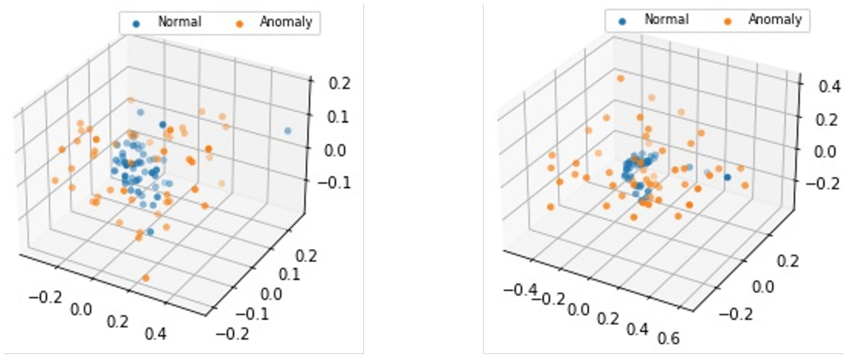


Figure 4.5: Predictions of the model on normal, augmented and anomalous samples from UCSD, MNIST, and CIFAR-10.



Class 8

Class 1

Figure 4.6: 3D visualization of learned representation of class 1 and 8 of the MNIST dataset. As shown, there are separable and distinct distributions of normal and anomalous embeddings.

Table 4.1: AUROC in % for anomaly detection on MNIST [25], Fashion-MNIST [61] and CIFAR-10 [24] datasets.

Dataset	Method	0	1	2	3	4	5	6	7	8	9	Mean
MNIST	AnoGAN[51]	96.6	99.2	85.0	88.7	89.4	88.3	94.7	93.5	84.9	92.4	91.3
	DSVDD[40]	98.0	99.7	91.7	91.9	94.9	88.5	98.3	94.6	93.9	96.5	94.8
	OCSVM[52]	99.5	99.9	92.6	93.6	96.7	95.5	98.7	96.6	90.3	96.2	96.0
	CapsNet$_{PP}$ [28]	99.8	99.0	98.4	97.6	93.5	97.0	94.2	98.7	99.3	99.0	97.7
	OCGAN[36]	99.8	99.9	94.2	96.3	97.5	98.0	99.1	98.1	93.9	98.1	97.5
	LSA[1]	99.3	99.9	95.9	96.6	95.6	96.4	99.4	98.0	95.3	98.1	97.5
	Ours (ADACL)	99.37	99.30	98.58	97.36	97.57	98.43	99.56	98.09	93.46	98.38	**98.01**
Dataset	Method	T-shirt	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot	Mean
Fashion-MNIST	DAGMM[69]	30.3	31.1	47.5	48.1	49.9	41.3	42.0	37.4	51.8	37.8	41.7
	DSEBM[65]	89.1	56.0	86.1	90.3	88.4	85.9	78.2	98.1	86.5	96.7	85.5
	LSA[1]	91.6	98.3	87.8	92.3	89.7	90.7	84.1	97.7	91.0	98.4	92.2
	DSVDD[40]	98.2	90.3	90.7	94.2	89.4	91.8	83.4	98.8	91.9	99.0	92.8
	OCSVM[52]	91.9	99.0	89.4	94.2	90.7	91.8	83.4	98.8	90.3	98.2	92.8
	Ours (ADACL)	94.42	99.46	89.82	91.05	92.68	90.40	80.43	97.88	97.14	98.88	**93.22**
Dataset	Method	Plane	Car	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck	Mean
CIFAR-10	OCSVM[52]	63.0	44.0	64.9	48.7	73.5	50.0	72.5	53.3	64.9	50.8	58.56
	CapsNet$_{PP}$[28]	62.2	45.5	67.1	67.5	68.3	63.5	72.7	67.3	71.0	46.6	61.2
	AnoGAN[51]	67.1	54.7	52.9	54.5	65.1	60.3	58.5	62.5	75.8	66.5	61.79
	DSVDD[40]	61.7	65.9	50.8	59.1	60.9	65.7	67.7	67.3	75.9	73.1	64.81
	LSA[1]	73.5	58.0	69.0	54.2	76.1	54.6	75.1	53.5	71.7	54.8	64.1
	OCGAN[36]	75.7	53.1	64.0	62.0	72.3	62.0	72.3	57.5	82.0	55.4	65.66
	CAVGA-D$_u$[56]	65.3	78.4	76.1	74.7	77.5	55.2	81.3	74.5	80.1	74.1	73.7
	DROCC[15]	81.66	76.74	66.66	67.13	73.62	74.43	74.43	71.39	80.02	76.21	74.23
	Ours (ADACL)	73.89	83.87	67.47	70.66	69.51	77.91	72.66	83.04	87.64	81.35	**76.80**

Table 4.2: Frame-level AUCROC and EER comparison % on UCSD dataset with state-of-the-art methods.

Method	AUCROC (%)	EER (%)
TSC [32]	92.2	-
FRCN action [18]	92.2	-
AbnormalGAN [38]	93.5	13
MemAE [13]	94.1	-
GrowingGas [55]	94.1	-
FFP [30]	95.4	-
ConvAE+UNet [33]	96.2	-
STAN [26]	96.5	-
Object-centric [20]	97.8	-
Ravanbakhsh [39]	-	14
ALOCC [45]	-	13
Deep-cascade [43]	-	9
Old is gold [64]	98.1	7
Ours (ADACL)	**98.4**	**5**

Table 4.3 compares our results with those of two other methods that use the knowledge distillation framework. Methods such as these rely on pre-trained networks which have been trained on millions of labelled images. To learn from these pre-trained or teacher networks, these methods have been trained over many epochs. These methods are computationally expensive and require a long time for inference, which prevents their use in real-world scenarios. Our method takes less time and computation to train even though the results are slightly lower as shown in the table.

Table 4.3: Comparison of AUROC in % for anomaly detection on MNIST [25] and CIFAR-10 [24] datasets with knowledge distilation methods.

Dataset	Method	0	1	2	3	4	5	6	7	8	9	Mean	Epoch
MNIST	U-Std[5]	99.9	99.9	99	99.3	99.2	99.3	99.7	99.5	98.6	99.1	99.35	-
	Multiresolution KDAD [48]	99.82	99.82	97.79	98.75	98.43	98.16	99.43	98.38	98.41	98.1	98.71	50
	Ours (ADACL)	99.37	99.30	98.58	97.36	97.57	98.43	99.56	98.09	93.46	98.38	98.01	10
Dataset	Method	Plane	Car	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck	Mean	Epoch
CIFAR-10	U-Std[5]	78.9	84.9	73.4	74.8	85.1	79.3	89.2	83	86.2	84.8	81.96	-
	Multiresolution KDAD [48]	90.53	90.35	79.66	77.02	86.71	91.4	88.98	86.78	91.45	88.91	87.18	200
	Ours (ADACL)	73.89	83.87	67.47	70.66	69.51	77.91	72.66	83.04	87.64	81.35	76.80	15

4.4.2 UCSD

To evaluate our method on video anomaly detection, we selected the UCSD dataset [8]. This dataset contains multiple outdoor scenes with mobile objects such as pedestrians, cars, wheelchairs, skateboards and bicycles. Frames with only pedestrians are considered as the normal class, while frames containing other objects are anomalies. This dataset contains two subsets named Ped1 and Ped2. Ped1 includes 34 training video samples and 36 testing video samples and Ped2 contains 2,550 frames in 16 training videos and 2,010 frames in 12 test videos with a resolution of 240 × 360 pixels.

Following [64], we follow a patch-based protocol to evaluate on this dataset where each frame is divided into 30 x 30 sections. For training, we include only patches that include pedestrians. We applied the previously mentioned data augmentations to each pedestrian patch to create anomalies for training. However, the model was evaluated on patches that contain pedestrians or other objects (At test time, no generated anomalies are being used). As evaluated in [64], to report the performance on this dataset, frame-level Area Under the Curve (AUROC) and Equal Error Rate (EER) are calculated in table 4.2. Results show that ADACL surpasses state-of-the-art methods for video anomaly detection on UCSD.

4.4.3 BCE vs. MSE

As part of this and the following sections we have done some complementary experiments as an ablation study. Based on our experimental results, we use MSE over BCE because it converges faster but still has good performance over multiple training runs. The Figure 4.7 shows that our experimental results are aligned with this hypothesis. Figure 4.7 compares the average of the test AUROC of the model on the CIFAR-10 dataset over 10 runs when using MSE versus BCE.

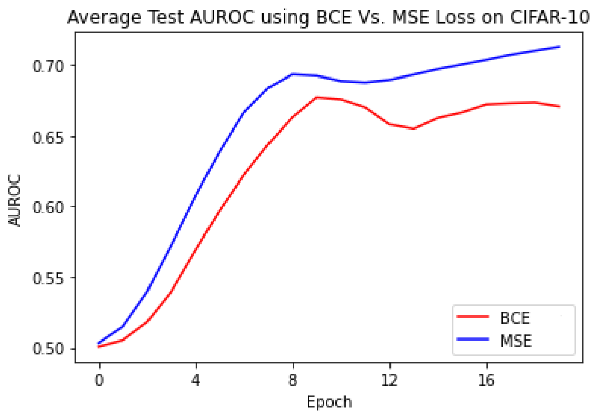


Figure 4.7: Average test AUROC taken over 10 training runs on CIFAR-10. MSE achieves higher AUROC in a shorter number of epochs.

Table 4.4: Experiments on different intervals

Interval	Mean AUROC (%)	Variance
$[0, 0.1] - [0.9, 1]$	97.14	$1.50 \times 10^{-}5$
$[0, 0.2] - [0.8, 1]$	96.93	$8.83 \times 10^{-}5$
$[0, 0.3] - [0.7, 1]$	97.42	$1.29 \times 10^{-}5$

4.4.4 Continuous vs. Discrete Labelling

Referring back to the method section, we define an early stopping criteria based on validation AUROC. In this experiment, we study how labelling affects anomaly detection performance on the CIFAR-10 dataset. We keep the entire training procedure the same and only modify the labelling scheme. Results show that discrete labels cause higher variance in the validation and test AUROC. With higher instability, it is increasingly difficult to create an accurate stopping criteria. As shown in Figure 4.8, continuous interval labelling yields lower variance in validation AUROC. Knowing this, Figure 4.9 shows that with continuous labelling, a stopping criteria over a validation set with lower AUROC variance is mostly able to produce more consistent test AUROC values per class. Therefore, continuous labelling is the better choice when using a stopping criteria for anomaly detection.

4.4.5 Effects of Augmentations

Variations in anomaly detection performance occur when using data augmentation to create anomalies. In this experiment, we study the effect of each augmentation used to create anomalies during training. In Figure 4.10, the performance of each solo augmentation is shown. As can be seen, the best performing solo augmentation is Cut-paste. But, the best anomaly detection results are achieved by using combination of all augmentations. This experiment is done on the MNIST dataset.

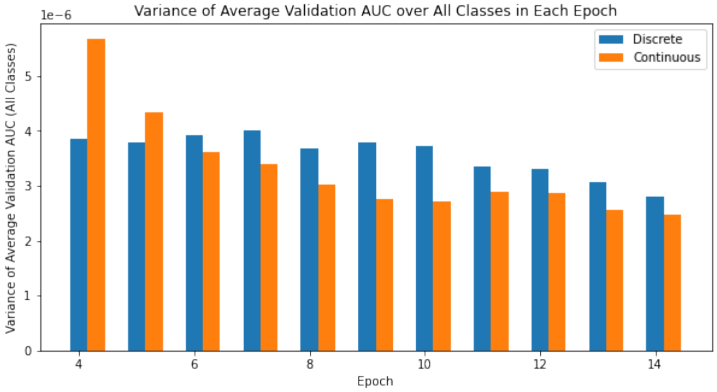


Figure 4.8: Variance of Average Validation AUROC over all classes in each epoch. The averages are taken over 10 training runs of CIFAR-10. It can be observed that continuous labelling consistently produces lower variance in higher epochs. Thus, a stopping criteria based on the validation set yields more stable test AUROC when continuous labels are used.

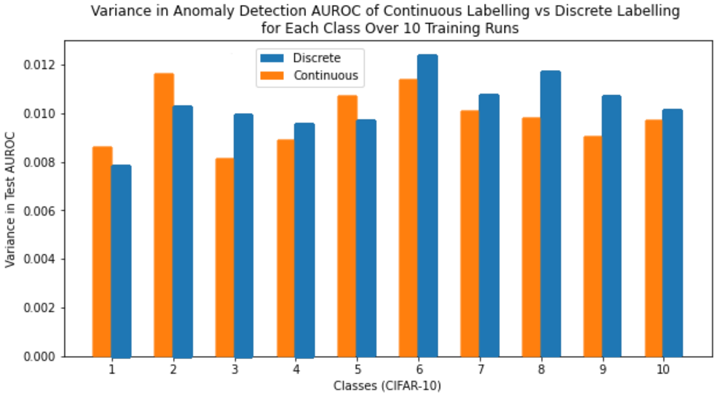


Figure 4.9: Variance in anomaly detection AUROC when using continuous labelling versus discrete labelling for all classes taken over 10 training runs of CIFAR-10. More often than not, using the same stopping criteria with continuous labels produces lower variance in test AUROC.

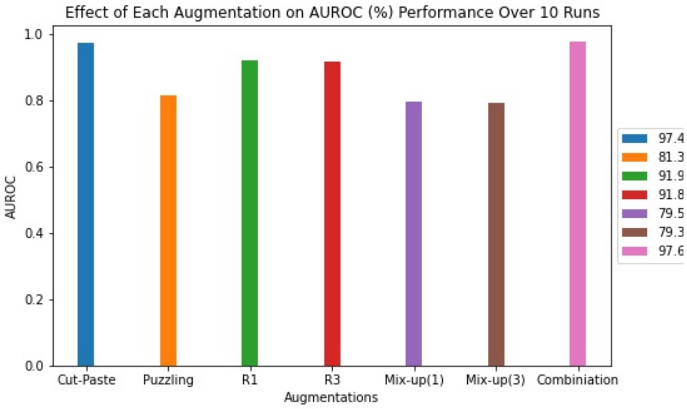


Figure 4.10: The effects of the augmentations like Cut-paste, puzzling, one and three times rotation and various mix-ups on AUROC. The most effective augmentation was Cut-paste, however using all augmentations in the process of creating outliers yields the highest performance in anomaly detection.

4.4.6 Interval Selection

In previous experiments, we show that continuous labelling improves anomaly detection by enabling better early stopping. This is achieved through lower variance in validation AUROC. To further examine the implications of label selection, We analyze different sized intervals when training the model on the MNIST dataset to see their effects on anomaly detection performance. As shown in table 4.4, the choice of interval has low impact on AUROC.

4.5 Conclusion

Deep neural networks can achieve state-of-the-art performance when applied to anomaly detection tasks. However, most of them suffer from expensive computations, high complexity and training instability. In this chapter, we alleviate these issues by proposing a simple and effective methodology for anomaly detection. We convert the problem into a supervised regression task by creating anomalies using data augmentations and training a lightweight convolutional neural network over continuous labels. In further experiments, we analyze the effects of MSE Loss versus BCE Loss, continuous labelling, interval size, and various augmentations. Results on several image and video anomaly detection benchmarks show our superiority over cutting-edge methods.

Chapter 5

Conclusion

5.1 Conclusion and Discussions

Anomaly Detection (AD) is to identify examples that vary in some respect from the training observations. These examples which do not conform to the distribution of normal data are called outliers. In real-world anomaly detection problems, the outlier class is absent, poorly sampled, or not well-defined. Hereupon, one-class classification methods are preferred to model this problem. Recently, anomaly detection draws huge attention and provide many applications in the computer vision field. In this thesis, we explored the limitations of current deep learning-based approaches for visual data and focused on solving two existing challenges.

The reconstruction error of Autoencoders (AEs) was shown to be useful in determining whether or not a sample follows the distribution specified in the training examples [60]. Yet, the main disadvantage of AEs is that they generalize very well to outliers and learn insufficient representations for the anomaly detection task. Chapter 3 of this thesis proposes a method to fix this problem. We propose an adversarial setting for identifying anomalies in both images and videos. Particularly, our method contains an encoder-decoder convolutional architecture that tries to reconstruct perturbed images and an encoder that attempts to generate effective perturbations from input data. This encoder is called Adversarial Distorter. Using the Adversarial Distorter, the reconstruction loss will be significantly increased, while the encoder-decoder network strives to reduce it. By perturbing the latent space of autoencoders, they can learn richer representations and perform better at anomaly detection. The results demonstrate that our method exceeds recent state-of-the-art models for anomaly detection.

Deep learning-based methods can accomplish state-of-the-art performance when applied to anomaly detection tasks. Nevertheless, most of them suffer from costly computations, high complexity, and training instability. In chapter 4, we ameliorate these issues by proposing a simple and practical method for anomaly detection. We transform the anomaly detection problem into a supervised regression task by creating anomalies using data augmentations. Then, we trained a simple convolutional neural network with continuous labels. In further experiments, we investigate the effects of MSE Loss versus BCE Loss, continuous labelling, interval size, and various augmentations. Results on several image and video anomaly detection benchmarks demonstrate our superiority over cutting-edge methods.

In conclusion, we researched the limitations of current deep neural network-based methods for anomaly detection in images and videos. We identified generalizability and complexity as two important challenges in current methods, and in this thesis, we propose two different approaches to address these two issues.

5.2 Applications

In general, anomaly detection is a task when the majority of the data is considered to be normal or there is no precise definition of anomalies. Anomaly detection has many applications in both industry and research. Applications of anomaly detection retain fraud detection in banking, fault detection in manufacturing, intrusion detection in computer networks, monitoring sensor readings in an aircraft. In computer vision, anomaly detection has widespread applications like medical problems in health data [51], monitoring video surveillance cameras [31] [54] and defect detection [48].

5.3 Limitations

Although both proposed methods are shown to be effective in their scope of comparison, there are some limitations for them. In chapter 3, we propose Adversarially Learned Perturbations (ALPS) method to solve the generalizability of Autoencoders (AEs). Even though ALPS is very successful to prevent AEs from generalizing to outliers, it is sensitive to the amount of perturbations added to the latent space or the strength of the adversarial distortion. We can either add coefficients to the loss for each network or train the Adversarial Distorter every couple of epochs to smooth out the effects of perturbations or the dominancy of the perturbation generator network during training. In the case of very complex video or image datasets, this method might be inapplicable.

In chapter 4, we propose Augment to Detect Anomalies with Continuous Labelling (ADACL) to alleviate the problem of complexity of some of the current methods by turning anomaly detection into a supervised problem by creating anomalies with data augmentation. Although ADACL has shown to achieve great performance on different anomaly detection benchmarks, it may not be able to perform defect detection. Defect detection is one type of anomaly detection in which the abnormality appears in a small part of the image. The reason for this issue is that the defects are very challenging to detect due to their size. In ADACL, we aimed to propose a method for detecting anomalies in general where the entire image is different from normal data, not specific to small changes in the texture of the image.

5.4 Future Research

In addition to addressing the two challenges we described earlier for anomaly detection, the current thesis could be continued to fix the limitations of the proposed methods. With ALPS, we can control the effects of added perturbations or change the convolutional architecture of the autoencoder to transformers [11] which were recently shown to be very effective for handling visual data. For ADACL, we can study more image augmentations and investigate their effect on anomaly detection performance. Evaluations of these new augmentations on defect detection tasks can be conducted to determine their effectiveness.

Bibliography

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

[2] Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data-generating distribution. *The Journal of Machine Learning Research*, 15(1):3563–3593, 2014.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[4] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[5] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4183–4192, 2020.

[6] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

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

[8] Antoni Chan and Nuno Vasconcelos. Ucsd pedestrian dataset. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(5):909–926, 2008.

[9] Yong Shean Chong and Yong Haur Tay. Abnormal event detection in videos using spatiotemporal autoencoder. In *International symposium on neural networks*, pages 189–196. Springer, 2017.

[10] Yang Cong, Junsong Yuan, and Ji Liu. Sparse reconstruction cost for abnormal event detection. In *CVPR 2011*, pages 3449–3456. IEEE, 2011.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[12] Mariana-Iuliana Georgescu, Antonio Barbalau, Radu Tudor Ionescu, Fahad Shahbaz Khan, Marius Popescu, and Mubarak Shah. Anomaly detection in video via self-supervised and multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12742–12752, 2021.

[13] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1705–1714, 2019.

[14] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

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

[16] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[17] Paul Hayton, Bernhard Schölkopf, Lionel Tarassenko, and Paul Anuzis. Support vector novelty detection applied to jet engine vibration spectra. In *NIPS*, pages 946–952. Citeseer, 2000.

[18] Ryota Hinami, Tao Mei, and Shin'ichi Satoh. Joint detection and recounting of abnormal events by learning deep generic knowledge. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3619–3627, 2017.

[19] Heiko Hoffmann. Kernel pca for novelty detection. *Pattern recognition*, 40(3):863–874, 2007.

[20] Radu Tudor Ionescu, Fahad Shahbaz Khan, Mariana-Iuliana Georgescu, and Ling Shao. Object-centric auto-encoders and dummy anomalies for abnormal event detection in video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7842–7851, 2019.

[21] John Taylor Jewell, Vahid Reza Khazaie, and Yalda Mohsenzadeh. Oled: One-class learned encoder-decoder network with adversarial context masking for novelty detection. *arXiv preprint arXiv:2103.14953*, 2021.

[22] Erland Jonsson, Alfonso Valdes, and Magnus Almgren. Recent advances in intrusion detection. *7th International Symposium, RAID 2004, Sophia Antipolis, France, September 15-17, 2004, Proceedings*, 2004.

[23] Youngjin Kim, Minjung Kim, and Gunhee Kim. Memorization precedes generation: Learning unsupervised GANs with memory networks. *arXiv preprint arXiv:1803.01500*, 2018.

[24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

[25] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[26] Sangmin Lee, Hak Gu Kim, and Yong Man Ro. Stan: Spatio-temporal adversarial networks for abnormal event detection. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1323–1327. IEEE, 2018.

[27] Chongxuan Li, Jun Zhu, and Bo Zhang. Learning to generate with memory. In *International Conference on Machine Learning*, pages 1177–1186. PMLR, 2016.

[28] Xiaoyan Li, Iluju Kiringa, Tet Yeap, Xiaodan Zhu, and Yifeng Li. Exploring deep anomaly detection methods based on capsule net. In *Canadian Conference on Artificial Intelligence*, pages 375–387. Springer, 2020.

[29] Wen Liu, Weixin Luo, Zhengxin Li, Peilin Zhao, Shenghua Gao, et al. Margin learning embedded prediction for video anomaly detection with a few anomalies. In *IJCAI*, pages 3023–3030, 2019.

[30] Wen Liu, Weixin Luo, Dongze Lian, and Shenghua Gao. Future frame prediction for anomaly detection–a new baseline. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6536–6545, 2018.

[31] Weixin Luo, Wen Liu, and Shenghua Gao. A revisit of sparse coding based anomaly detection in stacked rnn framework. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 341–349, 2017.

[32] Weixin Luo, Wen Liu, and Shenghua Gao. A revisit of sparse coding based anomaly detection in stacked rnn framework. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 341–349, 2017.

[33] Trong-Nguyen Nguyen and Jean Meunier. Anomaly detection in video sequence with appearance-motion correspondence. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[34] Hyunjong Park, Jongyoun Noh, and Bumsub Ham. Learning memory-guided normality for anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14372–14381, 2020.

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

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

[37] Masoud Pourreza, Bahram Mohammadi, Mostafa Khaki, Samir Bouindour, Hichem Snoussi, and Mohammad Sabokrou. G2d: Generate to detect anomaly. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2003–2012, 2021.

[38] Mahdyar Ravanbakhsh, Moin Nabi, Enver Sangineto, Lucio Marcenaro, Carlo Regazzoni, and Nicu Sebe. Abnormal event detection in videos using generative adversarial nets. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 1577–1581. IEEE, 2017.

[39] Mahdyar Ravanbakhsh, Enver Sangineto, Moin Nabi, and Nicu Sebe. Training adversarial discriminators for cross-channel abnormal event detection in crowds. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1896–1904. IEEE, 2019.

[40] 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.

[41] Lukas Ruff, Robert A Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. Deep semi-supervised anomaly detection. *arXiv preprint arXiv:1906.02694*, 2019.

[42] Mohammad Sabokrou, Mahmood Fathy, and Mojtaba Hoseini. Video anomaly detection and localisation based on the sparsity and reconstruction error of auto-encoder. *Electronics Letters*, 52(13):1122–1124, 2016.

[43] Mohammad Sabokrou, Mohsen Fayyaz, Mahmood Fathy, and Reinhard Klette. Deep-cascade: Cascading 3d deep neural networks for fast anomaly detection and localization in crowded scenes. *IEEE Transactions on Image Processing*, 26(4):1992–2004, 2017.

[44] Mohammad Sabokrou, Mohsen Fayyaz, Mahmood Fathy, Zahra Moayed, and Reinhard Klette. Deep-anomaly: Fully convolutional neural network for fast anomaly detection in crowded scenes. *Computer Vision and Image Understanding*, 172:88–97, 2018.

[45] Mohammad Sabokrou, Mohammad Khalooei, Mahmood Fathy, and Ehsan Adeli. Adversarially learned one-class classifier for novelty detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3379–3388, 2018.

[46] Mohammad Sabokrou, Masoud Pourreza, Mohsen Fayyaz, Rahim Entezari, Mahmood Fathy, Jürgen Gall, and Ehsan Adeli. Avid: Adversarial visual irregularity detection. In *Asian Conference on Computer Vision*, pages 488–505. Springer, 2018.

[47] Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, pages 4–11, 2014.

[48] Mohammadreza Salehi, Niousha Sadjadi, Soroosh Baselizadeh, Mohammad H Rohban, and Hamid R Rabiee. Multiresolution knowledge distillation for anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14902–14912, 2021.

[49] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. One-shot learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065*, 2016.

[50] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Georg Langs, and Ursula Schmidt-Erfurth. f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks. *Medical image analysis*, 54:30–44, 2019.

[51] 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.

[52] Bernhard Schölkopf, Alexander J Smola, Francis Bach, et al. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[54] Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6479–6488, 2018.

[55] Qianru Sun, Hong Liu, and Tatsuya Harada. Online growing neural gas for anomaly detection in changing surveillance scenes. *Pattern Recognition*, 64:187–201, 2017.

[56] Shashanka Venkataramanan, Kuan-Chuan Peng, Rajat Vikram Singh, and Abhijit Mahalanobis. Attention guided anomaly localization in images. In *European Conference on Computer Vision*, pages 485–503. Springer, 2020.

[57] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[58] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

[59] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[60] 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.

[61] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[62] Liang Xiong, Barnabás Póczos, and Jeff Schneider. Group anomaly detection using flexible genre models. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, 2011.

[63] Dan Xu, Elisa Ricci, Yan Yan, Jingkuan Song, and Nicu Sebe. Learning deep representations of appearance and motion for anomalous event detection. *arXiv preprint arXiv:1510.01553*, 2015.

[64] Muhammad Zaigham Zaheer, Jin-ha Lee, Marcella Astrid, and Seung-Ik Lee. Old is gold: Redefining the adversarially learned one-class classifier training paradigm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14183–14193, 2020.

[65] Shuangfei Zhai, Yu Cheng, Weining Lu, and Zhongfei Zhang. Deep structured energy based models for anomaly detection. In *International Conference on Machine Learning*, pages 1100–1109. PMLR, 2016.

[66] 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.

[67] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012.

[68] David Zimmerer, Simon AA Kohl, Jens Petersen, Fabian Isensee, and Klaus H Maier-Hein. Context-encoding variational autoencoder for unsupervised anomaly detection. *arXiv preprint arXiv:1812.05941*, 2018.

[69] 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.

Curriculum Vitae

Name:	Vahid Reza Khazaie
Post-Secondary:	BSc. in Computer Engineering 2015 - 2019 Shahid Rajaee University Tehran, Iran
Experiences:	Graduate Teaching and Research Assistant The University of Western Ontario January 2021 - April 2022

Publications:

1. **Khazaie, V. R.**, Wong, A., & Mohsenzadeh, Y. (2022). ADACL: Augment to Detect Anomalies with Continuous Labelling.

2. **Khazaie, V. R.**, Wong, A., Jewell, J. T., & Mohsenzadeh, Y. (2022). Anomaly Detection with Adversarially Learned Perturbations of Latent Space. In 2022 19th Conference on Computer and Robot Vision (CRV)

3. **Khazaie, V. R.**, Bayat, N., & Mohsenzadeh, Y. (2022). Multi Scale Identity-Preserving Image-to-Image Translation Network for Low-Resolution Face Recognition. Proceedings of the Canadian Conference on Artificial Intelligence.

4. Jewell, J. T., **Khazaie, V. R.**, & Mohsenzadeh, Y. (2022). One-Class Learned Encoder-Decoder Network With Adversarial Context Masking for Novelty Detection. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 3591-3601).

5. Bayat, N., **Khazaie, V. R.**, & Mohsenzadeh, Y. (2021, June). Fast inverse mapping of face GANs. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 2000-2004). IEEE.

6. Bayat, N., **Khazaie, V. R.**, Keyes, A., & Mohsenzadeh, Y. (2021). Latent Vector Recovery of Audio GANs with Application in Deepfake Audio Detection. Proceedings of the Canadian Conference on Artificial Intelligence.