Enhanced Anomaly Detection through Deep Feature Extraction

Leticia Robledo Díaz

4th Year Project Report
Artificial Intelligence and Computer Science
School of Informatics
University of Edinburgh

2023

Abstract

Anomaly Detection is a data-hungry domain which learns a target (or normal) concept to discriminate against all possible anomalies. Obtaining labelled datasets is a very expensive and time-consuming process, which is not scalable for large and long-term applications. Unsupervised Deep Anomaly Detection has exploded in the literature to tackle the Anomaly Detection problem. This project proposes an architecture that extracts the features of multiple related datasets to build robust target concept representations through Multi-Task Transfer Learning. These features are learned by deep unsupervised Anomaly Detection model f-AnoGAN, increasing its robustness against outliers on the MNIST, USPS and SVHN datasets. The proposed architecture's performance is evaluated against other state-of-the-art deep unsupervised Anomaly Detection models, and also against other Feature Extractors to investigate the effect of the different depths of Feature Extractors on the Anomaly Detection task, and the ability of this architecture to tackle multiple tasks.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Leticia Robledo Díaz*)

Acknowledgements

I would like to thank my project supervisor, Dr. Hakan Bilen, for introducing me to the topic of Anomaly Detection and for his expertise and advice throughout the development of this project.

I would also like to thank my family for all their constant support and encouragement throughout my degree. Finally, I want to thank my friends for making Edinburgh feel like home and all the memories I will cherish. Thank you.

Table of Contents

Chapter 1

Introduction

Anomaly Detection is a field of study focused on the automatic identification of the instances which deviate from an assumed concept of normality. These are called anomalies, and they range from irregular to rare, to erroneous or faulty behaviour. Anomaly Detection has been an active field of research for centuries, with the earliest records of "discordant observations" dating back to the 19th century [39]. It is widely used across industries, including statistics, bioinformatics, engineering, machine learning, cybersecurity and fraud detection [12]. Depending on the field of study, detecting anomalies can be crucial for decision making in many domains, like financial fraud or the detection of cancerous cells or tumors [39, 48].

Anomaly Detection algorithms aim to overcome two main challenges: generalizing correctly to unseen normal instances, and accurately discriminating against all possible anomalies. There are many approaches to Anomaly Detection based on their dependency to labelled data. The main disadvantage of label-dependant approaches is that anomalies are scarce, hence these methods require a vast amount of data, as well as manual feature engineering and labelling. The huge variety and scope of anomalies also requires an expensive anomaly generation process [39, 20, 48]. On the other hand, unsupervised approaches instead only learn the normal instances, reducing the vast amounts of data required and eliminating the manual labelling tasks. However, unsupervised models have been shown to have a higher rate of missed anomalies [20]. Enhancing unsupervised models is thus a very desirable task and an active area of research [20, 12, 39, 48].

The best-performing traditional unsupervised models include SVM [46], Isolation Forest [30], and CBLOF (Cluster-Based Local Outlier Factor)[22]. Some of these models have previously been enhanced by sharing knowledge from another domain, as shown by He et al.'s SVM model [15]. This enhancement technique, known as Transfer Learning, is based on sharing knowledge from a source domain to a target domain, increasing the model's generalization ability in the target domain. This is beneficial for Anomaly Detection because anomalies do not necessarily follow a unique data distribution, and a model trained on the target data domain may not necessarily recognise data from another. Using Transfer Learning enables the Anomaly Detection model adapt to new domains by pulling representations for the normal concept from another domain, enhancing its ability to detect unseen instances (novelties), and become more robust against outliers [49, 25, 15].

In addition, the advent of Deep Learning has produced a new family of more accurate models with improved generalization abilities have been developed to tackle the Anomaly Detection problem, surpassing the performance of previous non-deep unsupervised approaches [39, 20]. Some of these models include AnoGAN [45], DeepSVDD [40], VAE [37, 5] and DAGMM [51]. Despite their improved performance, they require a very large amount of data for very specific domain tasks, and are susceptible to over-fitting given their increased complexity. A poor fit to the normal data exposes the model to high rates of missed anomalies, setting these model's performance behind their expensive supervised counterparts [20, 39, 48].

This project proposes a model architecture that can improve the accuracy of a state-of-the-art deep unsupervised Anomaly Detection model. This is achieved through the simultaneous use of several related datasets to build robust concept representations. This enhancement technique, known as Multi-task Transfer Learning, enables the detector to adapt and generalise from different domains, becoming more robust to complex, out-of-distribution outliers [50]. This project hence proposes a model which leverages features from multiple datasets to enhance the performance of multiple tasks, improving the original model's performance. Multi-tasking learning has not yet been applied to state-of-the-art Deep models, despite the many benefits it presents. A deep unsupervised anomaly detection model which recognises the most essential features of the normal/target concept does not need to rely on vast amounts of target data, it is easier to train, and can be applicable to more than one task, breaking the single-task trend in the deep Anomaly Detection literature [20].

As such, this work presents an architecture which fulfils these advantages by decomposing the normal images into the most relevant, general and salient features, such that the deep Anomaly Detection models can learn robust representations of the normal concept. For instance, in bioinformatics applications, knowledge of normal and malignant features in other tissues can help, through the transfer of this knowledge, identify and classify unseen instances correctly. In this Computer Vision project three different digit datasets are used. Using semantically-related datasets helps bridge the domain gap[1] between normality concepts, enhancing the model's representation of normality by learning the most significant features. When applied to anomalies and normal images, the proposed deep unsupervised detector is able to differentiate between them with higher success rates than without the Multi-task Transfer Learning technique.

To achieve this objective, this project implements the combination of different architectures and datasets (domains) to enrich the feature representations and explore their impact on the selected deep unsupervised Anomaly Detection algorithm. The contributions of this project can be summarised as follows: the proposed architecture reduces the need for very vast amounts of data by using the multi-domain transferred features, it can be used for more than one Anomaly Detection task, breaking the trend in the literature, and most importantly, it increases the robustness of the state-of-the-art Anomaly Detection model against outliers.

[1]Difference between the source (training) domain and target (test) domain.

1.1 Goals and Research Question

Anomaly Detection methods are highly specialised, developed to learn a specific task in a specific domain [39, 20, 48]. This requires vast amounts of data to learn a complete, thorough concept of normality, and despite this, models can sometimes fail to generalize to unseen instances.

1. **Goal 1: Benchmarking and Model Selection**: ADBench recently proposed a benchmark on shallow Anomaly Detection methods across different supervision approaches, but there is no benchmark in the literature for deep unsupervised Anomaly Detection models [20]. This project will therefore build upon this work and benchmark the state-of-the-art unsupervised anomaly detection models on the selected datasets. This process will require a literature review, implementation and testing of these models.
2. **Goal 2: Model improvement**: This part will explore whether leveraging transfer learning and feature extraction can enhance performance of state-of-the-art Anomaly Detection, building upon the literature gap described in Chapter .

This project's goal can be decomposed into several Research Questions:

1. What is the best performing state-of-the-art model in this domain?
2. What is the impact of shallow and deep feature extraction?
3. Can a single anomaly detection model be applied to different tasks?
4. Does Transfer Learning benefit the generalization ability of the model?
5. Does feature decomposition decrease the rate of missed anomalies?

1.2 Contributions

This project's contributions can be summarised in the following points:

1. **Proposal of a new architecture that combines Multi-Task Transfer Learning with a state-of-the-art model**. This architecture introduces multi-task and cross-domain transfer learning to deep Anomaly Detection, analysing whether a single method can exploit this technique to overcome the current trend in Anomaly Detection which requires per-task specialization for effective detection.
2. **Benchmark of relevant Deep Unsupervised Anomaly Detection Methods**. Across the different Anomaly Detection perspectives, the most powerful and promising approaches were selected, implemented and benchmarked under equal circumstances to on different datasets to provide a fair comparison.
3. **Transfer Learning effect and comparison on f-AnoGAN**, using increasing levels of network depth to evaluate the degree of feature abstraction that improves model performance, contrasting results from the literature [34, 15]
4. **Increased model robustness against anomalies** by leveraging Transfer Learning of feature from the target domain.

Chapter 2

Background

2.1 Anomaly Detection Problem Definition

The problem of Anomaly Detection is formulated under the assumptions that anomalies are rare and different from the norm. To define this concept of normality, let $X \subseteq \mathbf{R}^D$ represent the data space of the Anomaly Detection task, then the concept of normality is defined as the distribution $\mathbf{P}^+$ on X. This distribution represents the ground truth concept of normality in this specific task. Any instance $x \in X$ is an instance of the normality concept, called an inlier. Inliers x_i map to a region in the learned probability distribution of the normality concept. Inliers include instances x_i which lie in the low probability regions of this distribution $\mathbf{P}^+$ on X despite deviating considerably from the learned law of normality. As shown in Figure 2.1, the normality concept of the image ”7” must include instances x_i which lie across all the probability regions of $\mathbf{P}^+$.

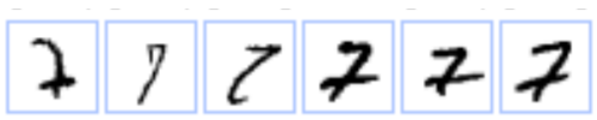

Figure 2.1: MNIST images of the number 7.

The task of Anomaly Detection algorithms is establishing the boundary between inliers and outliers, respecting the concept of normality while ensuring that novelties are also accepted. Novelties are instances of a probability region or mode of an evolving, non-stationary $\mathbf{P}^+$ [39].

An Anomaly Score is a numerical value assigned to a data point to measure its degree of normality. It is calculated with respect to a threshold and based on the presence or absence of features, such that a larger or smaller score indicates how much the data point deviates from the norm. Anomaly Scores enable differentiating between inliers in low-probability regions and anomalies.

2.2 Deep Learning: Concepts

Deep Learning is a subset of Machine Learning that enables the creation of models capable of learning from data to make accurate predictions. Deep Learning models leverage Artificial Neural Networks to make predictions or classification tasks. These Artificial Neural Networks (ANN) are inspired by human neural networks, and are composed of neurons assembled in layers and interconnected by weights, with each layer performing specific computations on the input data before passing it to the next layer. A network is composed of an input and output layer, with a variable number of hidden layers between them. The output layer uses the features that have been propagated through the network to perform classification or prediction tasks. In Anomaly Detection models, this is a binary classification task, predicting whether the instance is normal or anomalous[39]. ANNs can fit any continuous function to a high degree of accuracy. This makes them susceptible to overfitting, a problem which implies the network is memorising the training data rather than learning its features, and will fail to generalise well to unseen data instances [39].

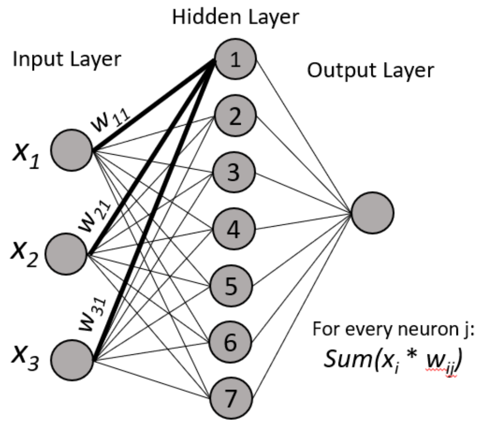

Figure 2.2: Simple Neural Network representation

The term ”Deep” in Deep Learning refers to the depth of the network, which has been shown to be beneficial as deeper networks learn more concise and complex data patterns [32]. Each layer in the network learns features at increasing levels of abstraction through vector operations with non-linear differentiable activation functions [32]. Gradient-based optimization methods like Stochastic Gradient Descent are used to compute the gradient of the error function with respect to the network’s weights, removing the need for manual feature engineering. Deep Learning approaches have proven to be successful in Anomaly Detection tasks like image classification, object detection and image segmentation through Convolutional Neural Networks (CNNs) [32, 39]. Deep Learning approaches have dominated the research output for Anomaly Detection, given the deep network’s ability to approximate complex distributions and non-linear functions through distinctive feature learning and identification [32].

2.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a type of neural network specialized for image and video processing tasks, but which have also shown impressive results in domains like Speech Recognition and Natural Language Processing[32]. These networks use convolutional layers to extract features from images by applying a set of learneable filters or kernels to the input data. These filters are small rectangular matrices of weights which slide over the input image pixels. They include parameters like stride[1] or padding[2] which determine how the filter slides (or convolves) over the image. This process is known as convolution, and it is what enables the convolutional layer to identify the presence of different features in different parts of the input [32].

Convolutional layers allow CNNs to learn hierarchical representations of input data, as the convolving is repeated throughout each layer of the network. The earliest layers learn edges and corners, and deeper layers learn higher-level features like object parts and complete objects as shown in Figure 2.3. CNNs have gained great popularity since the appearance of the first CNN, AlexNet [26]. CNNs in Anomaly Detection learn complex, intrinsic features of the normal concept at different levels of complexity, learning robust representations of the target domain. However, deeper networks are prone to the vanishing/exploding gradient problem, which can cause the model to get stuck in sub-optimal representations and fail to learn accurately (and thus generalize), or oscillate around the optimal solution or even diverge [9].

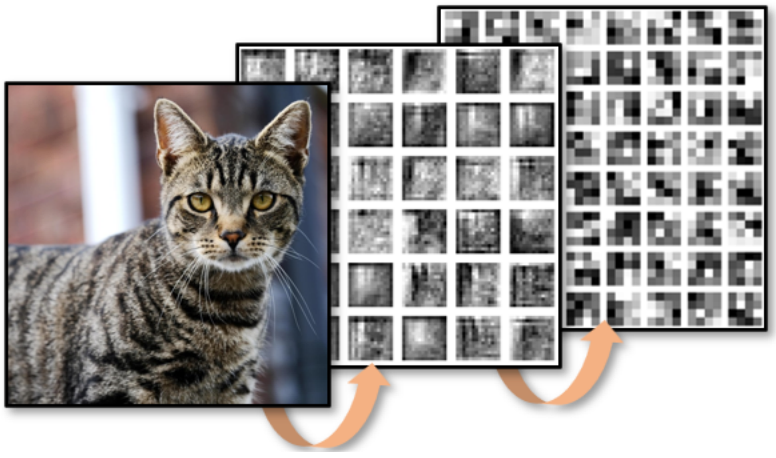

Figure 2.3: Hierarchical feature decomposition through feature learning in deep CNN. Left to right: input image, first layer and last hidden layer's feature maps of the image.

2.2.2 Neural Network Components

2.2.2.1 Fully Connected Layer

A neural network is composed of different kinds of layers, with the fully-connected layer's neurons being connected to all the neurons in the previous layer. Every neuron

[1]step size of the kernel/filter during the convolution operation
[2]skipped rows or columns before a convolutional operation

in this layer has a connection (weight) connecting to every neuron in the previous. Also known as fully-connected feedforward or dense layers, they are used in many neural networks to compute the weighted sum of input values.

They are powerful feature extractors and function approximators, as they adjust the weights and biases to approximate non-linear mappings like handwritten digit classification or speech recognition. In this project we will capitalise on this ability to create the feature extractors described in Chapters 4 and 5. Fully-connected layers can be susceptible to overfitting, and regularization techniques like weight decay or early stopping can be used to mitigate this issue [21].

2.2.2.2 ReLU Activation Function

The convolution and affine transformations of the network layers are linear, and as such learning non-linear patterns, can be achieved by through Activation Functions. Activation Functions enable deep learning networks to learn complex patterns like non-linear decision boundaries by introducing non-linearity to the network. Activation functions are added to the outputs of the network's hidden units, such that these non-linear transformations on the layer's outputs are fed into the next layer as inputs.

Common activation functions are ReLU, sigmoid or tanh. ReLU has been shown to perform better than sigmoid or tanh [17, 21] for an input x and a ReLU function f, this non-linear function returns the value $f(x)$ if this x is positive, 0 otherwise:

$$f(x) = max(0,x) \tag{2.1}$$

This provides several benefits to the training process, as it helps avoid the vanishing gradient problem during back-propagation of the network weights, while the sigmoid or tanh function squash the gradients and they approach zero. Assigning zero-values to the negative values introduces sparsity to the network, reducing the computational complexity and making training faster and more efficient.

2.2.2.3 Batch Normalization

CNNs contain several consecutive Convolutional Layers used to extract hierarchical representations of the input image. With activation functions adding non-linearity between the Convolutional Layers, Batch Normalization is used to normalize the input of each layer. The output of each layer is the input of the next, and the learned distribution can be altered as between layers. This is known as Internal Covariate Shift, and negatively affects the network's ability to learn as the distribution of input data is changed as it progresses through the network. This affects the network performance as the real distribution does not match the learned one.

Deep networks are prone to experience this problem, and thus models like ResNet18 include Batch Normalization layers after convolutional operations [21]. Batch normalization, through scaling and adjusting the activations, minimizes the Internal Covariate Shift, enhancing the network performance and improving convergence. Batch Normalization also speeds up training and improves generalization ability.

2.2.2.4 Pooling Layer

Once the convolutional filters are applied to the input image and features are extracted, the output of these filters is processed through pooling layers. Pooling layers reduce the size of each feature map while retaining the most important information, like the presence or absence of a feature in a specific region. Pooling layers have a ”window” hyper-parameter, which slides over the feature map and selects a subset of the pixels. Common pooling operations are Average Pooling, which takes the average of the pixels inside the pooling window as it slides over the feature map, or Max Pooling, which takes the largest pixel value. Pooling layers are crucial for the feature extraction process because they help control overfitting by reducing the complexity of the network parameters and reducing the size of the feature maps.

2.3 Deep Learning Techniques: Generative Modeling

Generative Modeling is an unsupervised learning task in the field of Deep Learning that learns to map samples drawn from a chosen distribution $\mathbf{Q}$ to the target distribution $\mathbf{P}^{+}$ as closely as possible. Generative Adversarial Networks (GANs) and Variational Autoencoders (VAE) are the most popular kinds of generative models.

2.3.1 Generative Adversarial Networks (GAN)

A Generative Adversarial Network (GAN) is a deep learning architecture which provides a way to learn detailed and realistic representations without extensively annotated training data [18]. This is achieved by deriving backpropagation signals as two networks, a Generator (G) and a Discriminator (D), are trained adversarially. During training, the Generator learns to generate realistic images that fool the Discriminator, while the Discriminator learns to differentiate the real data from the generated fake data. This process encourages the Generator to generate ”better” fake images by producing samples that approximate the target distribution. The Generator G maps a random noise vector z to the input samples, such that the Discriminator is not able to differentiate between synthetic and real images. Generative Adversarial Networks have been used in many fields, including image synthesis, image superresolution and classification [13]. They have been applied to Anomaly Detection through supervised (CatBoost [36]), semi-supervised (GANomaly [4]) and unsupervised (AnoGAN [45]) approaches.

2.3.1.1 GAN Challenges

GANs’ adversarial training objective, usually executed through an alternating optimization scheme, implies they are notoriously hard to train, mainly due to non-convergence in this adversarial (”min-max”) game [43], but also due to the Mode Collapse and Catastrophic Forgetting problems.

Fine-tuning a GAN can induce Catastrophic Forgetting, which appears where previously learned tasks are overwritten, and forgotten, by the learning of newer tasks. Regularization techniques or replay buffers are measures used to mitigate the issue [39].

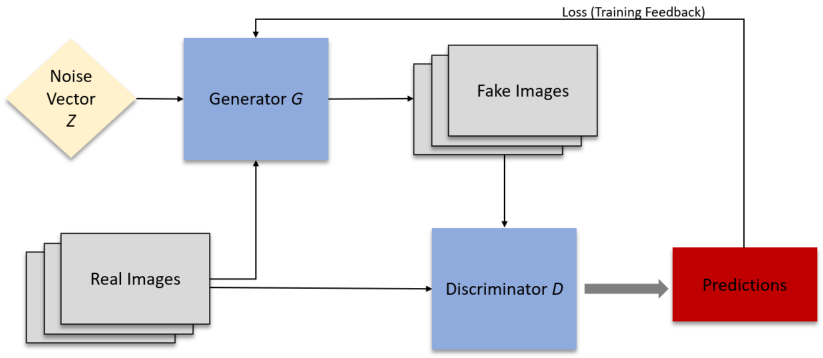

Figure 2.4: Generatve Adversarial Network Adversarial Training Process.

The Mode Collapse problem refers to the case where the Generator network learns how to fool the Discriminator by using a subset of the possible outputs, rather than learning to generate a diverse set of possible of outputs. The Generator has learned to generate competitive samples from a subset which is not representative of the entire dataset, limiting its generalization ability and overall usefulness. This problem can arise when the Generator created a fake sample that the Discriminator cannot differentiate from its real counterpart, and thus the Generator focuses on exploiting the Discriminator's weakness through similar samples. It can also be a symptom of poor hyperparameter tuning, lack of data diversity, or imbalanced networks, with one network overpowering the other one.

2.3.2 Autoencoders (AE)

Autoencoders learn by using two neural networks, an Encoder and a Decoder, to encode (and decode) compressed representations of the input data into a low-dimensional input space. They are an unsupervised deep approach which learns by minimizing the difference between the input data and the reconstructed representations. The encoder thus learns to extract the most relevant features and characteristics of the data in a meaningful way, and the decoder maps this compressed representations back to the input space, such that this difference between the reconstructed image and the real, input image is minimized. Autoencoders are common elements of deep Anomaly Detectors, like AnoGAN [45] or Variational Autoencoders [37, 5].

2.4 Transfer Learning: Multi-Task Feature Learning

Transfer Learning refers to the technique of transferring knowledge acquired from training a model in a source domain in a new target domain [49]. This knowledge is transferred through shared learned features, and can deliver better predictive performance [6]. Models trained in one domain tend to perform badly when evaluated in a different one; this is known as ”domain gap” [49]. A solution is Transductive Transfer Learning, which leverages knowledge from the target domain in the source domain for better domain adaptation [27].

Multi-task learning is a Transfer Learning technique which aims to maximize generalization through the identification of abstract and robust feature representations by simultaneously training on several different, related tasks [50]. Task relatedness has previously been modelled under the assumption that all the learned tasks' functions are close to each other in some norm, and can help bridge the domain gap [8, 16]. The simultaneous learning of tasks implies shared features enriching the learned concept representations. Leveraging these shared features can lead to better predictive accuracy than individual models trained on each task separately. This is beneficial in the domain of Anomaly Detection, as dynamic and evolving concepts of normality and limited data instances are frequent. Multi-task feature learning has been applied to several research fields, including computer vision, natural language processing and speech recognition [50].

2.4.1 Feature Extraction

A Neural Network can be divided into three fundamental elements, namely the input layer, the hidden layers, and the output layer, which performs the classification or predictive tasks with the features that the network extracts from the input image as it is propagated through the network as in Figure 2.2. The model has been trained to identify certain kinds of features. A neural network can become a Feature Extractor when, once trained on the target domain data, its final layer is removed or replaced, depending on the specific task. This enables access to the raw features called embeddings or encodings.

Transfer learning is very beneficial for a feature extractor's generalization ability. Systematic reviews on ImageNet-pretrained feature extractors have proven to enrich the model's feature representations [25, 41]. In this project, we will focus on Transductive Transfer Learning, a Transfer Learning technique which involves training the model in both the source and target domains, which is detailed in Chapter 3. This technique relieves the domain gap between datasets, providing better domain adaptation [27].

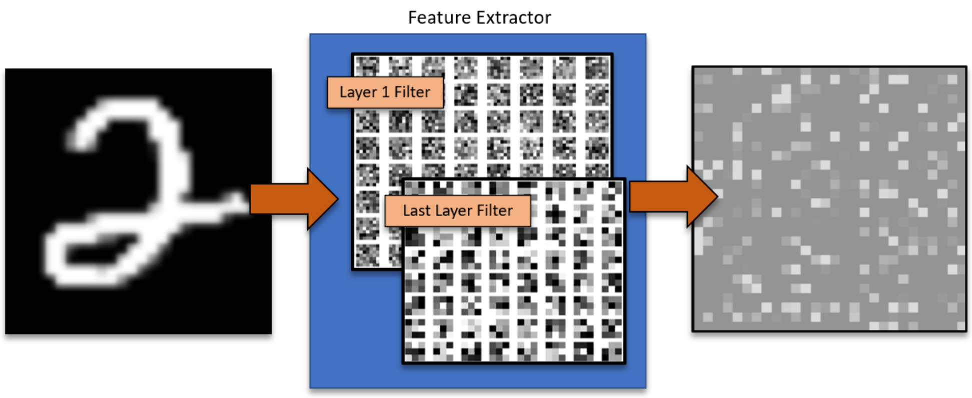

Figure 2.5: Illustration of the Feature Extraction Process

2.5 Critical Evaluation and Related Work

Anomaly Detection has seen many recent promising advances, but there is no domain performance comparison for the many multiple approaches. ADBench (Anomaly Detection Benchmark) recently published a benchmark of popular Anomaly Detection models, benchmarking them across domains and datasets, but their focus lies mainly on shallow models with distinct levels of supervision [20]. While this brings a useful reference point to the very ample Anomaly Detection domain, their benchmark is not inclusive of the latest, most complex and promising models. One of their conclusions is that each model should be evaluated to ascertain their suitability for the specific task, but this conclusion exacerbates the existing problem within Anomaly Detection: each model becomes highly-specialised on a specific task, requiring very large amounts of data for acceptable anomaly detection scores.

Ruff et al. present a review of the current Anomaly Detection paradigm, which shows that supervised and semi-supervised models require assumptions on the domain, data, prior knowledge about the anomaly distribution and about the specific application, such that the model becomes tailored to that specific problem and requires new (and expensive) data generation and manual labelling and feature engineering to adjust to the new normality concept [39, 2]. This presents two fundamental problems when using label-dependent approaches, namely the inherently difficult task of manual feature engineering to set the anomaly thresholds, and the limitation of label-dependant models to a static norm concept. These models also require an expensive ”maintenance” process, using dynamic manual thresholding and new data generation and labelling so the model maintains its predictive accuracy. Supervised models must also be monitored so it is not affected by the Catastrophic Forgetting problem (the model forgets previously learned concepts in the process of learning new ones) [39, 2]. These requirements are magnified with data-hungry deep methods, which have popularised due to their ability to capture complex data patterns [20, 39]

ADBench found that the expensive and time-consuming supervised Anomaly Detection process does yield models with respectable performance [20], but it is not sustainable or scalable for large-scale or long-term applications. In fact, some unsupervised models are able to surpass well-crafted label-dependant models in certain domains with a fraction of the resources, as unsupervised models only require data from the normal domain [20]. This work will therefore explore how to improve the more efficient unsupervised models on the chosen domain.

Furthermore, Ruff et al.’s survey on Anomaly Detection, in conjunction with Chandola et al.’s systematic review, show the crucial relevance of unsupervised models, and more specifically, of deep models given their powerful abilities to many complex capture patterns due to their increased parameter space [39, 12]. The Anomaly Detection domain can be divided into four main paradigms based on the strategy used to detect anomalies: Generative Adversarial Networks, Autoencoders, Gaussian Mixture Models, and traditional One-Class classifiers. The many existing models due to the single-task specialization trend in the Anomaly Detection literature, indicate there is no clear choice for any specific task. This project proposes to compare the performance of the most popular model within each category on the selected domain.

ADBench recently published a benchmark on shallow Anomaly Detection models, testing each algorithm on many different datasets [20]. This benchmark does not take into account the many latest deep models, which are complex and resource-intensive. This implies there is no overall consensus or benchmark dataset to compare them on; batch size and computing resources also greatly affect the benchmark experiments, and the trend in the literature is to dispute the previous model's findings and propose better-performing models [44, 40, 11, 15]. Bergmann et al. developed a benchmark dataset called MVTec to alleviate this issue, but it is mostly targeted towards the pixel-level and segmentation approaches of the AD paradigm [10]. This dataset is also lacking in diversity (only contains artificially-created anomalies, not representative of real scenarios), limited in size and lacking variability, which does not allow learning robust concept representations and limits the model's abilities. The datasets selected (Chapter 3) detail how to establish fair baseline comparison while accomplishing the multi-tasking goal of this project.

In terms of model selection, Schlegl et al. introduced the first GAN-based Anomaly Detection model for unsupervised data using a Deep Convolutionl GAN [45]. Advances in the GAN modelling techniques with the proposal of the Wasserstein GAN, f-AnoGAN was proposed as a fast, accurate Anomaly Detection technique [7, 44]. The appearance of GANs for Anomaly Detection were used for semi-supervised and supervised use cases, and new approaches like GANomaly were developed [4]. However, the latest developed approach, f-AnoGAN, has settled itself as the most promising for fast, convergent Anomaly Detection, which is especially relevant due to the lack of labelled anomaly detection data [44]. This approach is particularly powerful because it detects both pixel-level anomalies and whole image-based anomalies. This advantage breaks the trend of highly-specialised models, which this work aims to exploit further by applying the same model to different related tasks.

Another popular approach are Variational Autoencoders, which many Anomaly Detection use cases use with respectable results [5, 37, 31]. Furthermore, there are many one-class approaches due to the binary nature of the Anomaly Detection classification problem. Yang et al. propose an overview into the visual Anomaly Detection paradigm, which confirms that the main pitfall of these powerful deep, unsupervised models is their high missed anomaly detection rate [48]. These algorithm's anomaly scores are based on a Reconstruction Error, which can be influenced by patches within the image that have great similarity with the normal concept, and thus these anomalies map to a very close region within the normal probability distribution. This project explores whether identification of the image's most relevant features through shared knowledge can aid in the the classification of inliers and outliers.

The multi-task approach proposed in this work aims to leverage learned semantic similarity to generalise to unseen instances of the normal concept, extracting common features for each class by learning different dataset's representations of the same concept. Transfer Learning for visual Anomaly Detection has been successfully used before on single-task models, like Gopalakrishnan et al.'s VGG-16 ImageNet-pretrained CNN classifier for pavement distress [19], or Sabokrou et al. for crowded scenes [42]. This project adds to these literature findings by testing whether these results apply to the specific multi-task use case presented.

Chapter 3

Datasets

This chapter focuses on the dataset selection process and its relevance with respect to the Transfer Learning objective. It also describes the data batching system designed for even sampling across datasets, as well as all the preprocessing steps performed on each dataset.

3.1 Dataset Selection

The objective of this project, as described in Chapter 1, is to enable multi-task transfer learning to enhance performance of the task through Transfer Learning. Knowledge acquired by the network in related tasks is transferred to the current learning task. A multi-source domain strategy means the source data is composed of different datasets. The number of datasets learned can be interpreted as the number of learned tasks. In this case, multi-task feature learning requires that the source datasets share a small set of common features [6].

To maximize the benefits of transferred knowledge, datasets which were semantically and conceptually similar were selected. To accomplish this multi-task objective, data from the source and target domains should have the same size, and be compatible in their feature space. While unsupervised algorithms do not use the data's labels for learning, it was in the interest of model performance evaluation to select labelled datasets.

There exist few large, labelled datasets that fulfil the Transfer Learning objective of this project. Research into the literature shows that MNIST and CIFAR-10 are commonly used to evaluate overall model performance [39]. This project thus capitalises on the existence of a quality dataset such as MNIST and selects semantically similar datasets to exploit the Transfer Learning objective. Three datasets, MNIST, USPS and SVHN are thus selected, illustrated in 3.1. All selected datasets share 10 semantic classes, each corresponding to one digit from 0-9.

Data Split	MNIST	SVHN	USPS	Total
Train	60000	73257	7291	140548
Test	10000	26032	2007	38039

Table 3.1: Individual Datasets metrics, last column shows total number of images

3.1.1 MNIST Dataset

The MNIST dataset consists of 70000 images of handwritten digits from 0 to 9 [29]. Each grayscale image has dimensions of 28x28 pixels. MNIST is a popular benchmark dataset used to evaluate the performance of Deep Learning algorithms. These images are split into a training set comprising 60000 images, and 10000 images were allocated for the test set.

3.1.2 Street View House Numbers Dataset

The SVHN dataset contains 32×32 real-world, coloured (RGB) images of house numbers extracted from Google Street View [35]. This dataset comes from the significantly challenging and unsolved problem of recognizing digits and numbers in natural scene images. There are ten classes comprised of the digits 0-9, with the training set being composed of 73257 images and the test set of 26032. The original dataset presents images with character-level bounding boxes. The dataset format selected has been pre-processed such that the images are centred around each character. This format causes the target number to present other numbers or fragments of these at the sides (”distractors”), which are increases the difficulty of the digit identification task. This dataset has a very pronounced uneven class distribution, with smaller numbers being more common, especially the number ”0”, which has almost 14000 occurrences against the 5000 of number ”9” or ”1”.

3.1.3 United States Postal Service Dataset

This subset of the original USPS dataset is comprised of numerical digits that have been automatically scanned from envelopes by the U.S. Postal Service [24]. The images show centered and grayscale individual numbers from 0-9, with dimensions of 16x16 pixels. The training set has 7291 images, and 2007 images compose the test set.

3.2 Application to Anomaly Detection

The unsupervised deep learning algorithms evaluated in this project focus on identifying semantic anomalies, and thus the **Multimodal Normality** approach described in [39, 11] was adopted. The Multimodal Normality approach is based on two main assumptions, namely that anomalies are rare, and they are different from the norm. Consequently, the data is split such that the model is only trained on 80% of the selected target class for each dataset. The remaining training data, composed of 20% of target instances and the rest of the classes, were added to the test set, which includes an even distribution of all

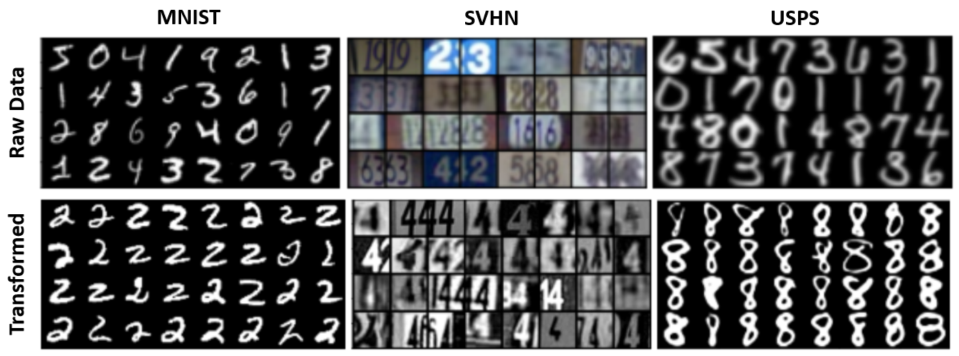

Figure 3.1: MNIST, SVHN and USPS dataset samples. Top row shows raw data format, bottom row the applied transformations

classes. The small subset of target instances removed from the training set is done to test the model's overall generalization ability by making the task more realistic.

This approach produces 10 different datasets from each of the 3 selected datasets, evaluating whether the model can correctly classify unseen inlier and outlier images for all the different classes across different domains (tasks). Figure 3.4 shows that the test sets of each class on the MNIST ans USPS datasets have a very similar number of samples, although this does not apply to some classes of the SVHN dataset, which will be taken into account when evaluating the algorithm's performance.

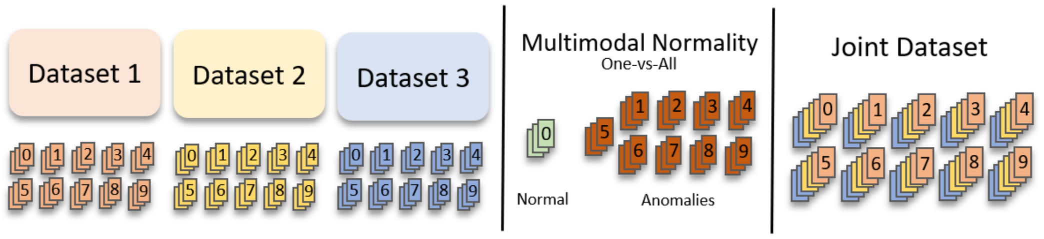

Figure 3.2: Multimodal Normality approach used for each number of each dataset, Joint Dataset used for training each Feature Extractor

3.3 Data Processing Steps

The three datasets share a common numerical domain but follow very different distributions. Most notably, the SVHN dataset is highly heterogeneous. Preprocessing operations were performed to enhance detectability [39]. To ensure that all the images from all datasets can share a label set and feature space, all images are re-scaled to size 28x28 using binomial interpolation, and are also normalized to have zero mean and unit variance. All dataset's samples are grayscaled so they have a unique dimension.

For the SVHN and USPS datasets, to enhance image quality and definition, both the contrast and sharpness of the images is increased. The USPS images are resized to dimensions of 28x28, while for the SVHN dataset, additional resizing from the 32x32 original dimension is done after center-cropping the image to a 29x29 dimension. This ensures that the surrounding numbers and distractors are excluded, or their presence minimized. The SVHN images are also transformed from RGB to grayscale, so all datasets have 1 single colour channel.

3.3.1 Batching system design

The datasets have a very different size and class distribution, with the combined datasets add up to 140548 samples. The individual dataset metrics and the combined total metrics are displayed in 3.1. To avoid the Catastrophic Forgetting problem and ensure an equal presence of each dataset's images during training, a batching system based on upsampling was developed, creating a merged dataset from MNIST, SVHN and USPS. This expanded the presence of MNIST, and especially of USPS, in each sample batch, as shown in Figure 3.3.

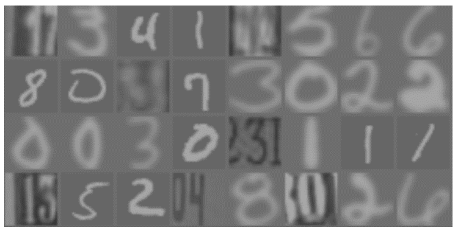

Figure 3.3: Sample batch of the Joint Dataset

The classes across the three datasets were aggregated, so each digit category includes instances with that same digit from all three datasets. Given the varying class size across datasets as shown in 3.4, the data from USPS and MNIST was upsampled to match the size of SVHN. This expanded the MNIST and USPS datasets through random re-sampling and shuffling to match the size of the SVHN dataset for each digit. This especially important for the USPS dataset since it has the least number of samples. The Training set contains a more pronounced uneven class distribution, which is not the case with MNIST or USPS, both of which contain even more equal distributions. which is uneven in the Training set, as shown by Figure 3.4

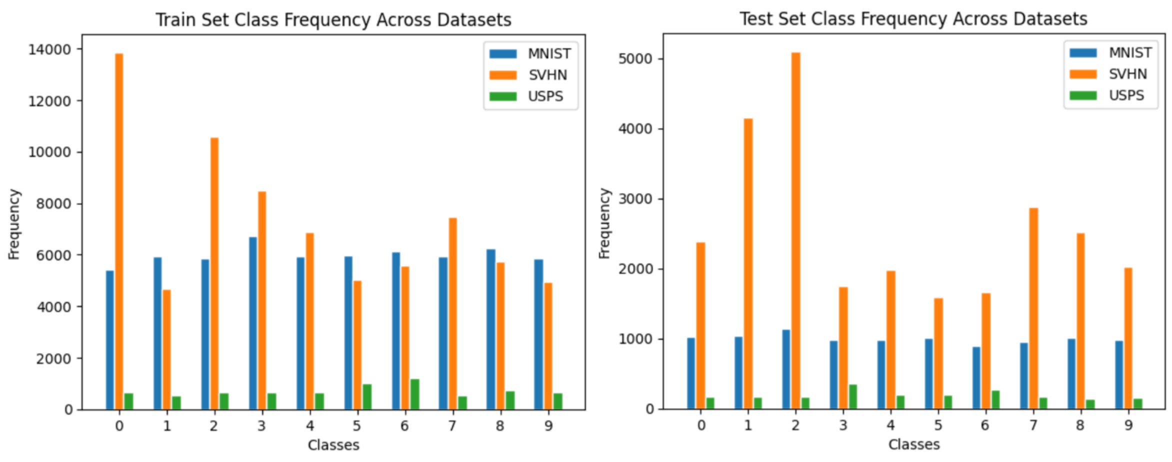

Figure 3.4: Class Distributions across Datasets

Chapter 4

Methodology

This chapter first describes the proposed architecture, and is followed by a two-stage approach: the first stage concerns the Anomaly Detection method selection process, and the second stage describes the Feature Extractor implementation and adaptation process. The outcome of all experiments which determine the final model is described in Chapter 5.

4.1 Proposed Architecture

The goal for this project is to improve upon the state-of-the-art anomaly detection through the use of Transfer Learning for feature extraction and enrichment from cross-domain knowledge. To achieve this, the architecture proposed in Figure 4.1 is presented. This architecture is composed of a Feature Extractor and a deep unsupervised Anomaly Detection algorithm. Each Task Head is specialised in a certain task to fulfil the multi-tasking objective. This Anomaly Detection algorithm is represented in Figure 4.1 as a Task Head, and showcases the cross-domain and multi-source knowledge transfer that the Feature Extractor enables.

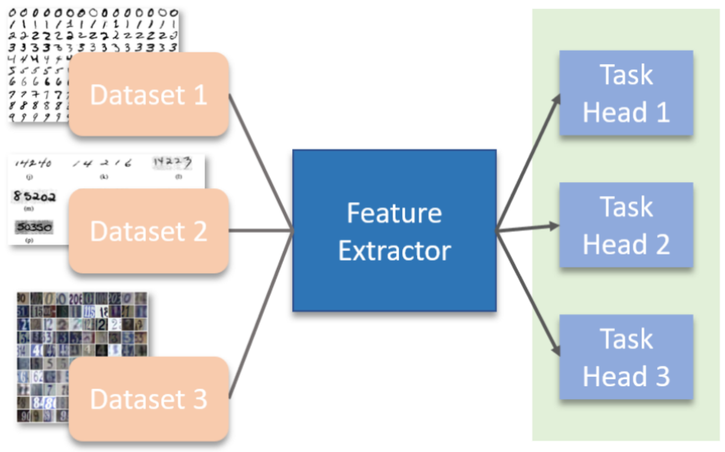

Figure 4.1: Proposed Architecture

This project can therefore be divided into two stages. The first stage is based on the identification of the most suitable deep unsupervised Anomaly Detection algorithm through benchmarking experiments under equal conditions. The second stage is based on the Feature Extractor model selection and implementation, which will be then combined with the Anomaly Detection model. The goal of this hybrid architecture is to achieve an improved robustness against anomalies than that of the unsupervised Anomaly Detection algorithm on its own.

4.2 Part 1: Anomaly Detection Model Benchmarking

This part of the project focuses on selecting the state-of-the-art unsupervised Anomaly Detection models described below, analysing their strengths and weaknesses, and explaining the approach for their fair, empirical comparison. Given their unsupervised nature, all algorithms are trained on normal, non-anomalous data. These models will be benchmarked in the first part of Chapter 5, and the best-performing model will be paired with the Feature Extractors described in Part 2 of this Chapter. Source code implementation details are described in Section 4.5. The four selected algorithms are:

4.2.1 Model 1: f-AnoGAN

F-AnoGAN is an improved version of the original AnoGAN, also proposed by Schlegl et al. and which introduced GANs to the Anomaly Detection domain [45]. The inherent structure and strategy of both f-AnoGAN and AnoGAN is based on an unsupervised training of an Autoencoder and a GAN, with the WGAN replacing the previous Deep Convolutional GAN of AnoGAN [38]. The algorithm fast-AnoGAN (f-AnoGAN) is a generative adversarial network (GAN) capable of identifying both anomalous images and anomalous regions within them [44].

The first component is a Denoising Autoencoder, which learns a robust representation of the data by mapping the images to a lower-dimensional latent space. It adds noise to the input data, and as it learns to reconstruct the original noise-free images, it removes both its added noise and other residual noise from the input samples [44].

The second component is a Wasserstein GAN (WGAN), which learns the distribution of the input data. The WGAN is a state-of-the-art GAN architecture developed by Arjovsky et al. to alleviate the problem of Mode Collapse and improve learning stability through convergence [7]. This algorithm learns the Generator G and Discriminator D networks adversarially, with the Discriminator estimating the Wasserstein distance; this distance is defined as the cost of the optimal transport plan for moving the (probability) mass in the predicted sample to match that in the target [7]. The Discriminator estimates the difference between the real and generated data's distributions, instead of directly discriminating between the real and generated samples.

This f-AnoGAN implementation follows the two-step training proposed by Schlegl et al., using gradient penalty to train the Generator G and Discriminator D, and discriminator-guided encoder `E` training using Schlegl et al.'s izi_f loss function. The gradient penalty is calculated using a random weight term α for interpolation between real and fake (generated) images:

$$interpolation = (\alpha \times real_samples + (1-\alpha) \times fake_samples) \tag{4.1}$$

The izi_f loss function are given by:

$$L_{izif} f(x) = \frac{1}{n} \times ||x - G(E(x))||^2 + \frac{\kappa}{nd} \times ||f(x) - f(G(E(x)))||^2 \tag{4.2}$$

During inference[1], the discriminator feature residual error that arises from mapping the input image to the latent space, as well as the image reconstruction error image from the GAN-generated image constitute the Anomaly Score A:

$$A(X) = (\frac{1}{n} \times ||x - G(E(x))||^2) + \kappa \times (\frac{1}{n_d} \times ||f(x) - f(G(E(x)))||^2) \tag{4.3}$$

The first term corresponds to the encoder loss, and the second to the discriminator-guided GAN training, with $\kappa = 1$ as presented in the paper.

This model was chosen because of its increased robustness to multi-modal data, faster and enhanced Anomaly Detection performance and the convergent nature of WGAN.

4.2.2 Model 2: Variational Autoencoders (VAE)

These Generative Architectures learn to represent dimensional data in a compressed, low-dimensional latent space using two neural networks, Encoder and Decoder. They learn a probabilistic mapping from the input data to the encoded representation, as the the Encoder compresses the data into a latent representation using the sample's mean and variance vector, which the Decoder reconstructs back into the original input space. This probabilistic interpretation of the latent space, different from the deterministic approach of regular Autoencoders, allows the model to learn the distribution of the encoded representation, rather than the point estimate of regular Autoencoders. This introduces variability in the generated data.

$$L(x,x') = -E[logp(x|z)] + KL(q(z|x)||p(z)) \tag{4.4}$$

Variational Autoencoder's Loss function is based on Reconstruction Loss to measure the difference between the original and reconstructed data. The Kullback-Leibler (KL) divergence acts as regularization term in the loss function, controlling the reconstruction fidelity's to the normality concept. During inference, the input sample is mapped to its latent representation using the Encoder network, and the Decoder uses this latent representation to reconstruct the sample. The Anomaly Score is computed from the reconstruction error between the reconstructed and original sample.

Variational Autoencoders were chosen because they are powerful and flexible in their sample generation, as they can generate data points similar to the normal samples because they learn through a probabilistic approach. By approximating the norm's distribution, VAEs can handle high-dimensional and complex data, making it a strong contender against other Anomaly Detection models.

[1]Inference refers to the process of using a trained model to make predictions on new, unseen data

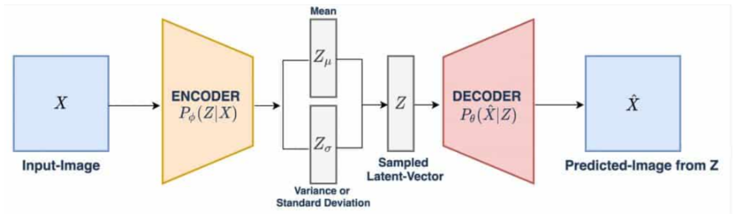

Figure 4.2: Variational Autoencoder Learning Process. Image sourced from [47].

4.2.3 Model 3: DeepSVDD

Deep One-Class Classifier (Deep SVDD) was developed by Ruff et al. with the aim of learning a neural network with an unsupervised Anomaly Detection objective, capable of learning a compact representation of the normal data such that anomalies fall outside this representation space [40]. This model was chosen due to its popularity and the robustness against outliers reported in some papers of the literature [40, 39, 48].

This algorithm's learning approach is based on two parallel objectives: training a neural network while minimizing a hypersphere that encloses the network representations of the data. This approach forces the network to extract common factors. Its Anomaly Score is calculated by measuring the distance to the center of this hypersphere, minimizing the average distance of all training samples to this point by using the weights W as regularization term[2]. The Anomaly Score A of the input data $\phi(x;W)$ mapped by the network ϕ with parameters W is this calculated by:

$$A(x) = min_w(1/n)\sum||\phi(x;W)-c||^2 + (\lambda/2)\sum||W||_F^2 \tag{4.5}$$

4.2.4 Model 4: DAGMM

Gaussian Mixture Models (GMMs) are a type of probabilistic model that can be used for anomaly detection. Gaussian Mixture Models model the data as a mixture of several Gaussian Distributions, where each distribution corresponds to a different cluster within the data [51].

A Deep Autoencoding Gaussian Mixture (DAGMM) model combines an Autoencoder with a Gaussian Mixture Model to estimate data density in a low-dimensional space. The Autoencoder learns a low-dimensional representation of the input data, using the traditional Autoencoder's Encoder and Decoder networks to map the data to the low-dimensional compressed space, as presented in Section 2.3.2. A Gaussian Mixture Model is then fit to the Encoder's low-dimensional input data representations, learning their distribution.

This algorithm's anomaly score is based on the negative log-likelihood of each data point's x_i compressed representation z_i, as shown in the equation below, where $p(k|z_i)$ is the posterior probability distribution of mixture component k given to z_i [51]. It is calculated based on the probability density under each Gaussian distribution, is its anomaly score, representing how unlikely it is for that data point to belong to any of

[2]This is a penalty term added to the objective function (Anomaly Scoring function) to prevent overfitting

the clusters. A smaller negative log-likelihood decreases the possibility of a data point being an anomaly [51]. This model was chosen due to its inherent multi-distribution structure.

$$E(z_i) = -log p(z_i) = -log \sum_k p(z_i; k) \tag{4.6}$$

4.3 Part 2: Feature Extractor Implementation

A Feature Extractor refers to the artifact or model able to extract the most relevant features from an input image. In this second part of the project, different approaches have been implemented to obtain feature embeddings of the input images. This part of the project focused on exploring the effect of feature extractors and whether they enable the Transfer Learning objective of this project. In this section, we will present the different Feature Extractors used, implementation approach, and methodology.

4.3.1 LeNet-5

This is a shallow neural network composed of 3 convolutional layers, 2 subsampling (pooling) layers and 2 fully connected layers as shown in Figure 4.3. This network, also known as LeNet-5, was proposed by LeCun et al., and remains a robust and computationally efficient due to the small number of learned parameters and small number of layers, but powerful enough to capture the underlying patterns in the data [28]. This combination of fast feature extraction through few convolutional layers enhances feature interpretability.

It was chosen because it is robust to overfitting, easy and fast to train, and less likely to memorize noise in the training data. The extracted features produce interpretable data embeddings since the input data is reduced to its most essential components, as its first two convolutional layers focus on detecting the edges and shape outlines, and the outer one detects the whole number shape.

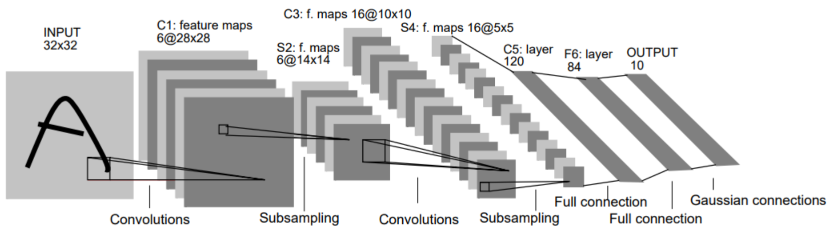

Figure 4.3: LeNet-5 Architecture proposed by LeCun et al. Sourced from [28]

The classic architecture of LeNet-5 passes the input data through the 3 convolutional layers, which contain pooling layers between them. These pooling layers use Average Pooling sub-sampling, which calculated the window pixel average value. The last two fully connected layers enable the model to perform classification tasks.

This architecture was modified to accommodate the downstream feature extraction task and the number of output channels in the third (and final) convolutional layer was increased to 144 from the original 120, as done in the literature [34]. This last layer connects to two fully-connected layers which compress the data to perform the classification of the numbers.

This modification increased the number of parameters and increased its learning ability, as it was observed it enhanced the performance on the Anomaly Detection task. This size was chosen because all input images to the Anomaly Detection algorithm should have the same square shape, to provide fair evaluation and prevent the detector from discriminating based on other artifacts like image shape.

4.3.2 Principal Component Analysis

This common technique reduces the dimensionality of data while retraining the maximum amount of variance [1]. The original features are transformed into Principal Components, which are linear combinations of the input image features. The principal components are eigenvalues and eigenvectors of the covariance matrix, which measures the linear relationships between pairs of features [1]. These principal components are ordered such that the first component captures the largest amount of variance in the data, and the last one explains the least.

4.3.3 ResNet18

ResNet is a deep neural network which was designed to address the issue of vanishing gradients that very deep models experience by introducing skip connections that allow gradients to flow more easily through the network, ensuring the model can learn and generalize better [21]. It was chosen due to its proven ability to capture complex patterns [21].

4.3.3.1 Original Architecture

It is composed of 17 convolutional layers divided into 4 consecutive convolutional blocks, as shown in Figure 4.4. Each convolutional block is composed of two convolutional layers, followed by a batch normalization layer, and a ReLU activation function. The input data flows from each block, with each block transforming the data and abstracting its features further. Skip connections are inserted between the two convolutional layers of each block, and they are added element-wise to the output of the first convolutional layer and the result is fed to the second convolutional layer. Skip connections prevent the vanishing gradient problem, and help preserve important features throughout the network, as information from previous layers flows to deeper layers. Batch normalization and pooling layers compress the image features, maximizing the ability of the neurons to extract meaningful feature maps [21].

4.3.3.2 Adapted Architecture

The three datasets contain inherently different distributions and feature spaces. The ResNet18 architecture was modified to ensure that the network performed the multi-task learning and focused on the identifying the features representative of each digit class.

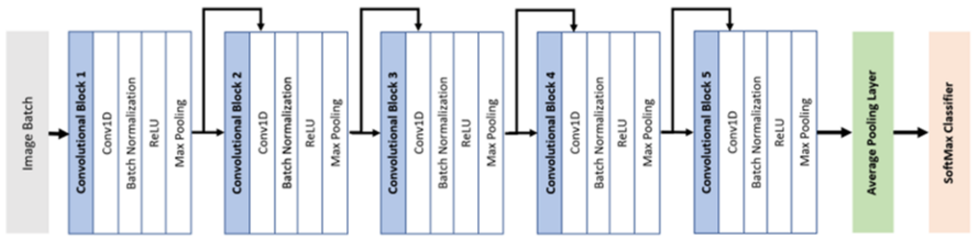

Figure 4.4: Original ResNet18 Architecture

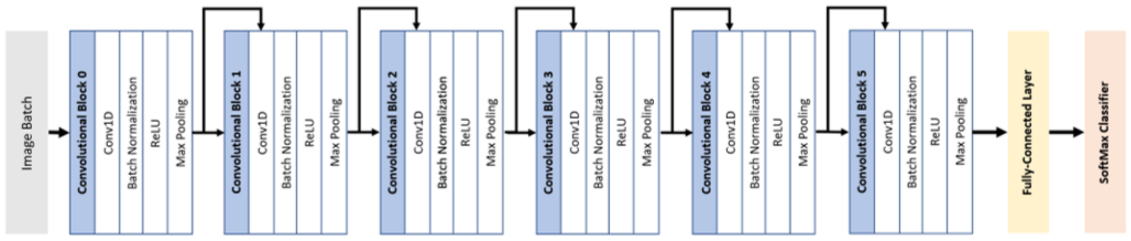

Figure 4.5: Adapted ResNet18 Architecture: Additional Convolutional Block (Block 0) and Fully-Connected Layer

The input layer, Convolutional Block 0, is a Convolutional Layer that expands the 1-dimensional, grayscale multi-source input feature vector to 64 dimensions, retaining the same parameters and other components as the original ResNet18 blocks. This was done because it was observed that adding this block reduced the noise in the SVHN samples, improving the classification on SVHN but not negatively affecting that of USPS or MNIST. A Max Pooling down-sampling operation is done before entering the first of the convolutional blocks (Convolutional Block 1). The network input and output dimensions are thus adapted to (1,64), (64,64), (64,128), (128,256), (256,512), (512,10) for the 5 blocks and fully connected layer, respectively, as shown in Figure 4.5. The fully-connected final layer at the network's output is trained to approximate a 512-dimensional image representation into a 10-category classifier (given the 10 categories in the Joined Dataset). The original Average Pooling layer is skipped, because the input images (28x28) are already very small at this point of the network and due to the down-sampling performed as they pass through the network great information loss is be experienced.

4.3.4 ResNet50

This network is a deeper version of the ResNet18, increasing the network parameters from almost 11.7 million to 25.6 million. This augmented complexity enables the model to capture more succinct and elaborate patterns within the data. This network was selected to evaluate the effect of more elaborate feature extractors on the downstream task. Given the nature of the ResNet architecture as presented by He et al., this network is composed of more convolutional blocks with interleaving residual connections [21].

This network was adapted to include a single Convolutional Layer (not a Block like in the ResNet18 model), with 1 input channel, a convolutional kernel (filter) has value of 7 and stride of 2, padding the images with 3 pixels on all sides. This ensures input consistency with ResNet18. At the output layer, no layers were removed, and instead the same strategy of adding a Fully-Connected Layer was used, approximating the high-dimensional feature space to the 10 digit categories.

4.3.5 From Classifier to Feature Extraction

The Neural Networks were transformed into feature extractors by first training the model on all three datasets to predict the 10 digit classes correctly, using the fully-connected layers to approximate the features from the last convolutional layer to each of these classes. The model parameters and weights are saved and the fully connected layers are then replaced by Identity layers using PyTorch's `nn.Identity()` module. These Identity layers enable the features to pass through the network without additional transformations, such that the network now outputs a feature vector for every input image the network.

4.3.5.1 Model Connection

The three approaches presented in Chapter 3.2 were adopted as feature extractors. The neural networks, namely the LeNet-5 and ResNet models, had their feature maps from the previous pre-training step loaded into modified architectures, which replaced the last 10-class classification layer by an Identity layer I. This implies that the network's high-dimensional output feature embeddings M were fed directly to the Anomaly Detection model using matrix multiplication operations for a regular matrix M_R as model input :

$$M \times I = M \times R = M_R \tag{4.7}$$

The transformations R performed on the feature vector M varied for each feature extractor given the different output vectors. These are the transformations applied to each network's output feature vector:

1. In the case of the **LeNet-5 network**, the output feature embedding matrix was a 1-dimensional 144 feature vector, which was resized to a 12×12 feature embedding.
2. In the case of the **ResNet network**, the output has 512 features. The Identity layer I replaces the feed-forward down-sampling layer, and the transformations matrix R creates a two-dimensional 16×16 feature vector M_R. Figure 4.6 showcases the feature vectors generated for numbers 1, 4 and 9.

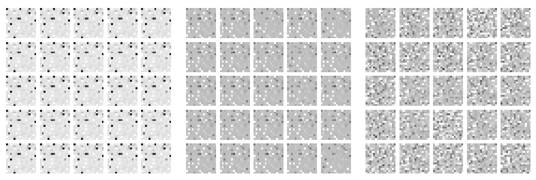
Figure 4.6: Output Feature Vectors for several target digits (left to right): 1, 4, 9

4.4 Evaluation Metrics

The evaluation of a model is crucial to understand its performance with respect to other models, and therefore it is important to use meaningful and universal evaluation metrics. In Anomaly Detection, choosing the appropriate evaluation metrics is essential since model suitability and performance for specific applications can change drastically depending on the evaluation metric considered [39, 48]. Let following terms are defined according to the Anomaly Detection use case:

- True Positives (TP) are the anomalous instances correctly identified as anomalies
- False Positives (FP) are the normal instances incorrectly classified as anomalies
- True Negatives (TN) are the normal instances that are correctly classified as normal
- False Negatives (FN) are the anomalies that are incorrectly classified as normal instances

Precision, Recall and AUC Score are standard metrics for the evaluation of Anomaly Detection Algorithms, such that:

- **Precision**: Percentage of correctly identified anomalies with respect to the total number of anomalies. A precision score of 1.0 (or 100%) indicates the model perfectly identifies the anomalies under all circumstances. A high precision indicates a low False Positive (FP) Rate, that is, there is a low rate of normal instances that have been incorrectly flagged as anomalies.

$$Precision = TP/(TP+FP)$$

- **Recall**: Percentage of correctly identified anomalies with respect to the total number of anomalies. A Recall rate is crucial to understand the robustness of the model against simple (easily identified) and complex outliers (anomalies which follow the normal samples' distribution).

$$Recall = TP/(TP+FN)$$

- **Area Under the ROC Curve (AUC)**: this measure evaluates the performance of anomaly detection models under a broad range of possible scenarios. The AUC Score, sometimes referred to as AUROC Score, measures the effectiveness of a binary classification model by plotting the true positive rate (TPR) against the false positive rate (FPR) at different decision thresholds (scenarios). Its values range from 0 to 1, with a score of 0.5 indicating random guessing, and a score of 1 indicating perfect performance. This area will be reported as a discrete metric consisting of the area under the curve (AUC) of the false positive rate and true positive rate, summarising the performance of the model with respect to the missed and identified anomalies, respectively.
- **Accuracy**: Percentage of correctly classified samples with respect to the total number of samples. This metric is used to evaluate the Feature Extractor's

ability to predict the different classes they are trained on, providing a fair model evaluation since the datasets are not highly unbalanced.

$$Accuracy = (TP + TN)/(TP + TN + FN + FP)$$

The goal for a robust Anomaly Detection model is high True Positive Rate and high precision, with a very low False Positive Rate. In some contexts, the False Positive Rate might be the most important target metric, since missed anomalies can be more costly than missed normal samples [39]. A high precision but low recall implies that the model consistently identifies a small number of samples correctly, but there are many missed anomalies. Conversely, a low precision and high recall rate implies that the model's outlier threshold is too high, and the False Negative Rate is too high (which implies the model is flagging both normal and anomalous samples as anomalies). Traditional performance metrics like Mean Absolute Error are not appropriate for this specific use case given that the models use a variety of scoring functions as pertain each algorithm.

4.5 Adaptation of Code and Challenges

The Deep Learning Anomaly Detection are not as common and popular as their shallow counterparts due to the large data and computational resources required to train them. This implies scarce code references and open-source implementations. The implementation of F-AnoGAN proposed by Schlegl et al. [45] is available on Tensorflow[3], but its use required re-writing in PyTorch due to library issues and incompatible dependencies. I used open-source code for initial implementation setup[4] and the Wasserstein GAN architecture[5].

`PyOD`[6] is a Python Outlier Detection library which was developed to facilitate the benchmarking efforts of ADBench by providing Python-based implementations of Anomaly Detection algorithm implementations [20]. This is not an exhaustive library, and its unsupervised algorithms are mostly based on shallow anomaly detectors. This library was used to implement the DeepSVDD and Variational Autoencoder algorithms. DAGMM used the original author's approach, using PyOD's GMM for initial inspiration.

The benchmarking part of this project required a considerable amount of time, as it entailed understanding the existing codebase and implemented algorithms, troubleshooting the imports and broken dependencies, creating their corresponding non-conflicting environments and translating the f-AnoGAN algorithm. This process was also lengthened by the considerably long compute time required by these algorithms.

For the Feature Extractors, PyTorch was used to implement the LeNet-5 network, following the original paper [28]. For the ResNet18 and ResNet50 networks, the PyTorch source code on the `vision` GitHub repository was adapted and transformed, using the corresponding ImageNet pre-trained weights for some of the experiments[7].

[3]https://github.com/tSchlegl/f-AnoGAN
[4]https://github.com/A03ki/f-AnoGAN
[5]https://github.com/eriklindernoren/PyTorch-GAN
[6]https://pyod.readthedocs.io/en/latest/
[7]https://github.com/pytorch/vision/blob/main/torchvision/models/resnet.py

Chapter 5

Experiments and Results

Anomalies differ from the concept of normality, and these can vary from out-of-distribution, highly detectable instances, to complex and difficult to identify samples. To simulate this real-life scenario, each experiment (except those that use the joint dataset) uses the Multimodal Normality approach presented in Chapter 3, such that one target number, simulating the concept of normality, is selected for each experiment and the model trained on it; the resulting model is evaluated using an unseen subset of the target number alongside all the other numbers in order to evaluate its generalization abilities.

Each section in this chapter aims to achieve each objective defined in Chapter 1. The first section presents the results of the model benchmarking experiments, and culminates with the selection of the best deep unsupervised Anomaly Detection model. Section 2 includes the experiments performed on the Feature Extractors. Finally, Section 3 will showcase the results and discussion of the combination of these Feature Extractors and the selected Anomaly Detection model. The code used to generate the tables and figures, as well as the data, experiment parameters and summary of results is in the Appendices of this paper.

5.1 Benchmark Experiments

Given the unsupervised nature of these algorithms, they are only trained on the target class of each individual dataset, with the rest of the classes labelled as ”outliers”, using the specific train/test data partitioning technique explained in Chapter 3. At the input layer, the target images were loaded in raw format with `torchvision` and PyTorch's `DataLoader` with batch size 64[1]. The different datasets are modified using `transforms.Compose()`, applying the multiple respective transforms to each dataset, such that each batch is composed of 28×28 1-channelled images.

The Multimodal Normality approach implies that each algorithm was trained on one target number of each dataset, so a total of 10 experiments for each dataset were run;

[1]Different batch sizes were tested, and although larger batch sizes improved performance on some models, they presented large computational constraints in the training of others

with three datasets (MNIST, SVHN, USPS), 30 different experiments were performed on each algorithm. This results in 120 different tests for the Baseline experiments: 10 digits $\times 3$ datasets $\times 4$ epoch settings = 120 experiments. To understand the fit of the models to the data better, each experiment was run a different number of epochs, namely 20, 100, 150 and 200 epochs, respectively. This yields a total number 120×4 epoch settings = 480 experiments. Each experiment was run with 3 different seeds, increasing the total number of experiments to 1440 total runs.

Model	Dataset	AUC	Precision	Recall
f-AnoGAN	MNIST	79.67 ± 0.84	98.04 ± 2.13	51.07 ± 0.84
	SVHN	55.73 ± 0.52	89.98 ± 2.86	50.32 ± 0.52
	USPS	90.24 ± 0.70	96.05 ± 0.26	51.69 ± 0.69
VAE	MNIST	78.82 ± 6.05	94.72 ± 0.06	55.51 ± 6.05
	SVHN	48.75 ± 0.29	96.78 ± 1.44	37.42 ± 0.29
	USPS	79.53 ± 7.40	96.79 ± 0.37	57.35 ± 7.39
DeepSVDD	MNIST	66.30 ± 5.74	95.40 ± 0.38	50.38 ± 5.74
	SVHN	48.91 ± 1.00	91.05 ± 1.31	40.82 ± 1.10
	USPS	71.43 ± 7.71	96.19 ± 0.48	55.34 ± 7.72
DAGMM	MNIST	60.44 ± 3.67	94.05 ± 0.16	64.94 ± 3.67
	SVHN	51.02 ± 3.93	92.06 ± 1.14	61.04 ± 3.93
	USPS	51.58 ± 5.00	93.11 ± 0.73	51.13 ± 5.00

Table 5.1: Average performance metrics on each dataset across all epochs.

5.1.1 Evaluation of Results

The baseline experiments for each algorithm on each dataset across all digits and epochs are summarised in Table 5.1. The AUC score can produce very optimistic results despite the very low recall in case of highly imbalanced test sets; this is the case with the test set used, which has an approximate ratio of 1:9 for the target:anomaly class distributions [14, 3]. For this reason, the AUC score's components, Recall and Precision, are used to further evaluate model performance.

Precision scores and the recall values are very different, following a consistent pattern of low recall and high precision. As shown in Section 4.4, a high precision value shows the model's identified anomalies are indeed anomalies, while the recall value indicates the proportion of identified anomalies with respect to the total number of anomalies. These results show that all algorithms have learned the concept of normality well enough that it enables the model to consistently flag certain anomalies (high precision), but this behaviour does not extend to all anomalies (remaining digits).

Recall scores are worse for the SVHN dataset across algorithms, but particularly for DeepSVDD and VAE. DeepSVDD uses a distance-based anomaly score to determine the *anomalousness* of each sample, and it is therefore unsuitable when detecting multiple types or very complex anomalies which can lie near the hypersphere center. The Variational Autoencoder performs considerably worse on SVHN than on MNIST and USPS, which is a consequence of its encoding of the sample's statistics into the latent space, which makes it less effective for highly variable data like SVHN, which has multiple modes of normality given its intra-class variability, this resonates with Nalisnick et al.'s findings which demonstrate that neural generative models like VAE assign higher likelihood to out-of-distribution samples, decreasing its effectiveness in anomaly detection tasks [33].

The DAGMM model achieves the highest recall rate on MNIST and SVHN. This model learns non-linear low-dimensional representations, which the Gaussian Mixture Model component can use to learn complex and multi-modal data distribution. This explains the high recall rate, as the anomalies are mapped to a latent space and their reconstruction mapped back to one of the multiple learned Gaussian distributions, producing a higher out-of-distribution mapping rate which successfully singles out more anomalies.

Model	Dataset	AUC	Precision	Recall
f-AnoGAN	All	77.59	97.52	52.69
VAE	All	49.28	97.40	49.28
DeepSVDD	All	46.14	97.32	46.15
DAGMM	All	54.35	93.36	59.18

Table 5.2: Average performance scores of each algorithm on the test set of each dataset across all epochs

With respect to the AUC Score for all algorithms on MNIST, f-AnoGAN achieves the best performance and DAGMM the lowest. The average AUC scores of all algorithms on all datasets are compared in Table 5.2. f-AnoGAN has a higher AUC Score for all SVHN, MNIST and SVHN than any other model as shown in Table 5.1.

The results show that, overall, the best-performing Anomaly Detection model on these three datasets with the best AUC (AUROC) score is f-AnoGAN, despite the higher average recall score of DAGMM. The DAGMM model has a lower precision score than f-AnoGAN, which implies that while it is able to detect a slightly larger number of anomalies, it fails to detect variations of it with the same frequency as f-AnoGAN.

In this work, the aim is to maximize the overall performance, that is, a high recall rate that shows the model capturing the largest number of anomalies, while maintaining a high precision value that guarantees that the same anomaly is consistently recognised. Ruff et al. argue that a higher precision value enables a higher concentration of anomalies with lower rate of alarms, also ensuring that the anomalies it does "catch" are consistently identified [39].

Based on this reasoning, the selected model to improve upon therefore was f-AnoGAN, using DAGMM's scores as reference.

5.1.2 f-AnoGAN: Error Analysis

Inspecting f-AnoGAN's performance, the left plot in Figure 5.1 shows the different AUROC scores for f-AnoGAN on all the different datasets across the different training epochs. While the AUROC scores mask the very low average recall score per epoch, it

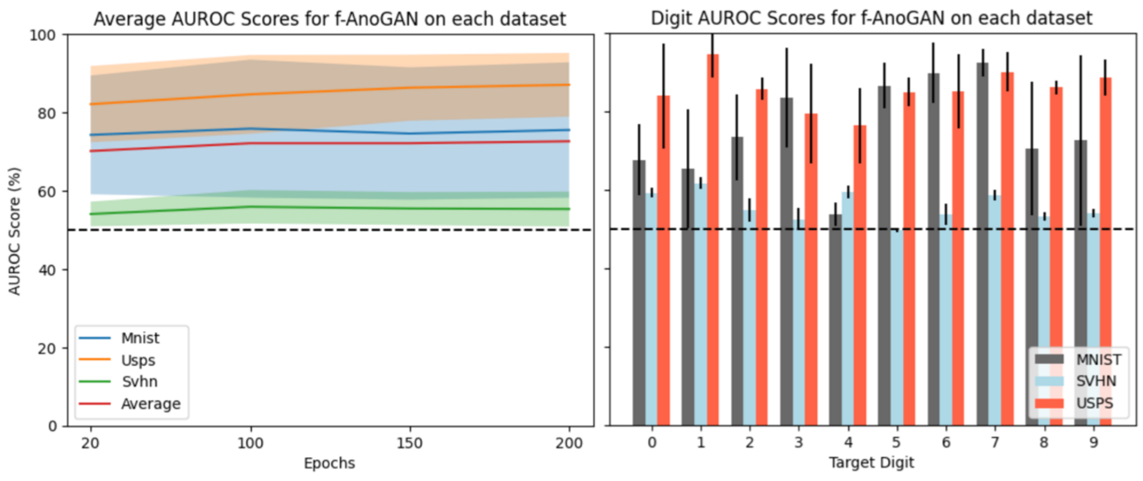

Figure 5.1: **Left**: f-AnoGAN AUC Scores across epochs on MNIST, SVHN and USPS. **Right**: f-AnGAN AUC Scores when trained on each target digit, for all datasets

also indicates that the common features and regularity within MNIST and USPS yields a better performance overall. It also shows how the number of epochs has a noticeable improvement on the USPS dataset, but this effect is not appreciated on the MNIST or SVHN datasets beyond 100 epochs. The performance on the SVHN dataset is a barely better than the random guessing baseline.

Visualization of the performance of f-AnoGAN on each individual digit category through the right plot in 5.1 shows the average performance of this algorithm (averaged across all epochs) when trained on each target number of each dataset. An AUC (AUROC) score of 0.5 (50%) is a baseline for random predictions. The average AUROC score across all epochs was considered alongside the standard deviation to visualize the overall performance. The visualization of Digit AUROC scores shows that performance of f-AnoGAN when trained on each across numbers is not uniform, and that the model finds it harder to generalize from certain numbers than others. The model performs considerably worse when trained on the SVHN dataset, across all digits, barely exceeding the random 0.5 baseline, especially for digits 3, 5 and 8. This behaviour is also appreciated in the MNIST dataset with the digits 0, 1 and 4.

In terms of specific digit performance, the model performs better digits 1 and 7 across all datasets, as the SVHN's poor average performance performs slightly better. These classes are the seocnd and third most common within the SVHN test set (depicted in Figure 3.4 in Chapter 3 3. This shows the model is able to clearly learn the fundamental features of these two numbers, and the model is able to clearly yielding an improved detection performance.

5.2 Feature Extractor Experiments

The neural networks introduced in Section 4.3 were trained to accurately predict all digits. The Joint Dataset described in Chapter 3 was used for this purpose, such that each model is trained on all datasets simultaneously, with the digit classes aggregated and using a batch sampling approach as described in Chapter 3. The pre-processing steps performed and described in Chapter 3 force the three datasets to share a more similar feature space.

The models learn the features through backpropagation of the weights. The train dataset was split 80/20 into training and validation sets to monitor the fit of the network to the data. Both validation and test sets were unseen by the network. Each feature extractor was trained on a different number of epochs, and the best performing model was saved. As each model contains a different number of parameters and varying complexity to learn the distributions of the three different datasets, experimentation with the batch size and the number of epochs is performed on all models.

The training process ensures the model learns the appropriate feature maps to correctly identify, extract and leverage the key features that differentiate the input images [8]. In this section, the experiments showcase the exploration process on how to exploit this ability and leverage this knowledge for the feature extraction task.

5.2.1 Training the LeNet-5 Network

With a batch size of 64 and a Cross Entropy Loss function, this model achieved an optimal 96.15% accuracy on validation data, and 90.20% on test data as shown in Figure 5.2 at 20 epochs. The model fits the data quite well throughout, but overfits the data beyond 20 epochs, as the validation and test set accuracy scores start decaying. The model is evaluated on validation and test data with static (frozen) weights, so there is no back-propagation of the calculated score upon the model weights. The best-performing model is on epoch 20, since it presents the best validation and test set fit.

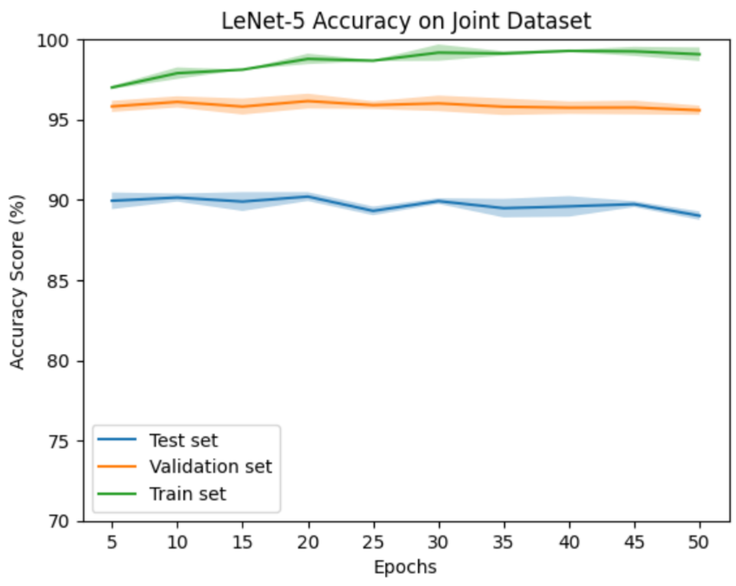

Figure 5.2: LeNet-5 model accuracy on train, validation and test sets

5.2.2 Training the ResNet18 Network

This architecture was trained on the Joint Dataset using a batch size of 32 to maintain batch heterogeneity while avoiding that each batch does not overwrite/undo the statistics calculated in the previous batch, inducing Catastrophic Forgetting (Chapter 2), remaining within the necessary computational limits. Early stopping was adopted to avoid overfitting. Two different approaches were used:

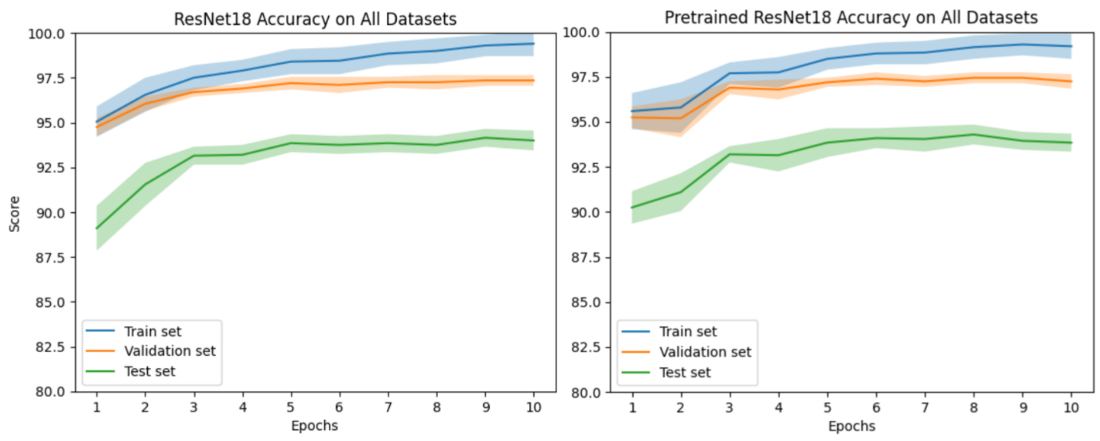

Figure 5.3: **Left:** ResNet18 model accuracy on train, validation and test sets. **Right:** ResNet18 pretrained on ImageNet accuracy on train, validation and test sets.

1. **Pre-trained on the Target Domain:** The architecture only uses target domain information to create its feature maps. As shown in Figure 5.3, this model achieves an optimal test score of 94.10% on the 9th epoch, coinciding with the optimal validation score of 97.49%. The model starts to fit the data well early in the training process, at around the 5th epoch, as the accuracy on the train set increases over 99% while validation and test set plateau beyond 5 epochs. The model chosen as feature extractor is the one at 9 epochs. This ensures that the model has very accurately learned the features of each category, generalising across their domains.

2. **Pretrained on ImageNet:** The ResNet18 architecture used ImageNet pre-trained weights. The network was fine-tuned on the multi-source source dataset. This implies the network's feature maps, learned through ImageNet, were re-learned to fit the data. The model achieves the highest validation accuracy with 97.55% at the 8th epoch, and the best test set accuracy at the 6th epoch with 94.30% accuracy. The validation accuracy in this epoch is 97.38%. This model was chosen as the feature extractor given the trade-off between best test set and slightly decreased validation score.

Both models performance plots show the models fit the training data to very high accuracy, while the test set data does not achieve the other sets' accuracy rates. This could be explained by the inherent diversity of the three different datasets, especially from the SVHN dataset.

5.2.3 Training the ResNet50 Network

This network was trained on using a 64-batch size and early stopping, halting the training on the 30th epoch due to the validation and training accuracy not increasing. The training set accuracy achieves almost-perfect accuracy performance, with 99.84% at the 27th epoch. Validation set accuracy at this epoch is 97.55%, and test set score is 94.67%. This epoch achieves the best performance across all data sets, and the weights pertaining to this epoch were saved for the Feature Extraction task.

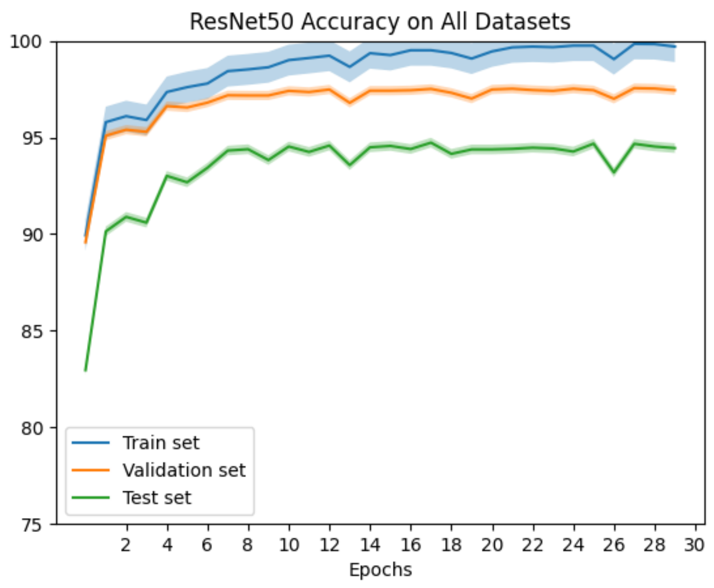

Figure 5.4: ResNet50 performance on the Joint Dataset

5.3 Multi-Tasking Model Experiments

This section will explain the experiments and how proposed model is able to learn multiple tasks from the source domain and generalise to a target domain. This section presents the results of the experiments of the different feature extractors on the best-performing model f-AnoGAN. The feature extractor and f-AnoGAN combination model was run on 5 different seeds and following the same strategy adopted in the benchmarking experiments5.1. This section will discuss the reported results of each individual model, followed by an in-depth analysis and discussion with respect to the project's research questions presented in Section 1.1[2].

Goal 2: Improvement of state-of-the-art Anomaly Detection method.
The Experimental results shown in Table 5.3 illustrate the best results obtained for the original model, and the results obtained through the model combinations. The second row shows the results of the ResNet-18 + f-AnoGAN hybrid multi-tasking

[2]ResNet18 + f-AnoGAN was also run without pretrained weights (no knowledge of target domain or ImageNet), but results were too poor to provide competitive discussion.

Model	Dataset	AUC	Precision	Recall	AUC Avg
f-AnoGAN	MNIST	79.67 ± 0.84	98.05 ± 2.13	51.07 ± 0.84	75.21 ± 0.69
	SVHN	55.73 ± 0.52	89.98 ± 2.86	50.32 ± 0.52	
	USPS	90.24 ± 0.70	96.05 ± 0.26	51.69 ± 0.69	
ResNet18 + f-AnoGAN	MNIST	76.74 ± 3.24	97.33 ± 2.34	54.14 ± 3.34	82.65 ± 2.02
	SVHN	84.63 ± 1.04	97.08 ± 1.37	57.72 ± 2.04	
	USPS	86.58 ± 1.79	96.20 ± 4.81	52.54 ± 1.80	
ResNet18 + f-AnoGAN (ImageNet)	MNIST	78.94 ± 0.72	97.16 ± 2.29	53.39 ± 1.72	76.01 ± 1.04
	SVHN	79.10 ± 1.04	95.01 ± 2.91	57.77 ± 1.94	
	USPS	69.99 ± 1.36	95.84 ± 4.09	52.38 ± 1.36	
ResNet50 + f-AnoGAN	MNIST	69.67 ± 0.91	97.18 ± 2.71	50.45 ± 0.91	70.29 ± 1.39
	SVHN	74.99 ± 1.02	97.75 ± 2.44	51.01 ± 1.02	
	USPS	66.22 ± 2.23	92.63 ± 8.34	50.56 ± 2.23	
LeNet-5 + f-AnoGAN	MNIST	50.67 ± 0.53	98.15 ± 0.68	51.06 ± 0.53	49.92 ± 1.05
	SVHN	48.46 ± 0.48	95.48 ± 2.17	49.97 ± 0.48	
	USPS	50.64 ± 2.14	97.75 ± 2.44	50.22 ± 2.14	

Table 5.3: Average performance scores of each algorithm on the testset of each dataset across all epochs.

model, demonstrating that this new architecture exceeds Schlegl et al.'s f-AnoGAN generalization ability on all three datasets: MNIST, USPS and SVHN.

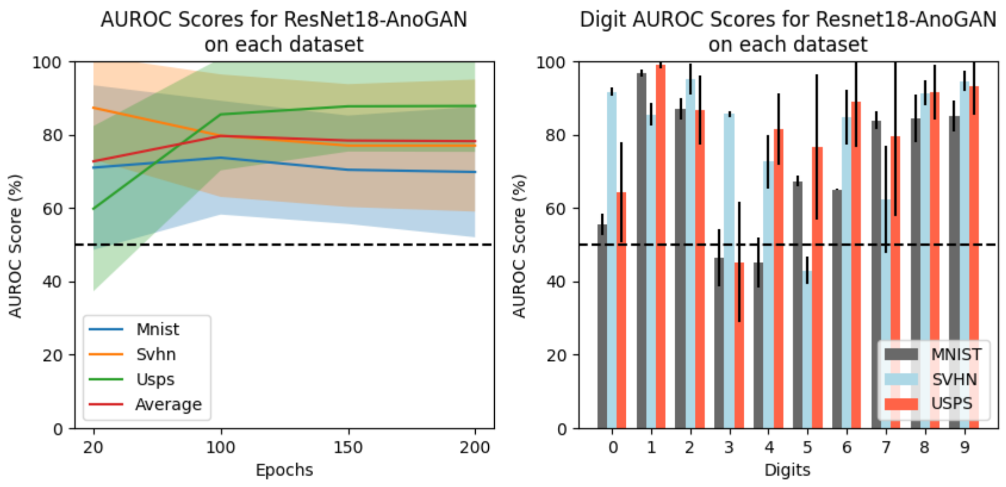

Figure 5.5: Multi-tasking hybrid architecture ResNet18 + f-AnoGAN average performance for all each dataset across epochs (left), and average performance for each digit (right)

Figure 5.5 show the AUC scores for each digit across epochs, such that each digit was trained as a target (normal) class for this ResNet18 + f-AnoGAN model. This model's precision on the SVHN dataset jumps from 89.98% to 97.08%, attaining almost-perfect identification with the highest average precision score reaching up to 98.45%. The Recall score on this dataset also achieves almost an 60% value.

These results indicate that using ResNet18 pretrained on the Target Domain as a Feature Extractor enables f-AnoGAN to overcome the high variability between samples, but it is now also able to detect a wider variety of anomalies. The incorporation of a feature extractor thus increases the robustness against outliers through an improved generalization ability. The AUC Score is precision-dominant, and as such the improved generalization ability upon MNIST is not as noticeable, despite correctly detecting up to 58.78% of all the total anomalies. This confirms Research Questions 4 and 5, which will be explored in depth in the following Sections.

5.3.1 Result Analysis: Transfer Learning and Generalisation Ability

The introduction of domain adaptation through Transductive Transfer Learning is very successful using ResNet18, as the results of both ResNet18 models showcases the benefits of shared, unified conceptual features, as shown in Table 5.3. However, visualization of the results in Figure 5.5 shows that the performance on each digit is not uniform, suggesting that the benefits of transferred knowledge are leveraged differently on each domain, with the SVHN benefitting the most from the regularity of the MNIST and USPS datasets. This confirms one of the Research Questions (RQ4), such that unified and robust concept representations can be beneficial to improve performance not only on different domains, but for different tasks within a same domain, given the consistently high performance of the SVHN dataset across digits.

Furthermore, ImageNet-pretrained weights and target domain transferred knowledge through the pretrained ResNet18 + f-AnoGAN model provides a slightly improved generalization ability when compared to the f-AnoGAN model. These benefits are very similar as the previous model's, but experience less variability (smaller standard deviation scores) (as shown in Table 5.3). This can be explained because ImageNet is much larger than the Joint Dataset 3, and as such it has already learned to recognize many different objects and features across many computer vision tasks. Transfer of this knowledge for domain adaptation leads to more consistent and reliable results, as the ResNet model has a better understanding of the underlying patterns in the data. It could be argued that pre-training on ImageNet provides a better domain adaptation approach than solely training on the Target Domain.

5.3.2 Error Analysis: ResNet18 + f-AnoGAN

Figure 5.5 shows that the performance on the SVHN dataset has greatly improved, while that of MNIST has not experienced the same benefit, as compared with Figure 5.1. Given that the performance of the model on each dataset is not uniform across epochs unlike the original f-AnoGAN model's, the optimal performance is chosen at the 100th epoch (Figure 5.5). Visualization of the AUROC Curves of all digits in the SVHN Dataset showcases the ResNet18 + f-AnoGAN's's correctly detected anomalies (True Positive Rate) against the missed ones (False Positive Rate) for each number, when each number is trained as a target (normal) instance.

As shown in Figure 5.6, the original model's performance shows is barely above the random guessing baseline (0.5), with almost half of the total anomalies missed. The best-performing architecture, ResNet18 + f-AnoGAN, achieves improved results, with the rate of correctly identified achieving an almost-perfect score of 1 for most digits.

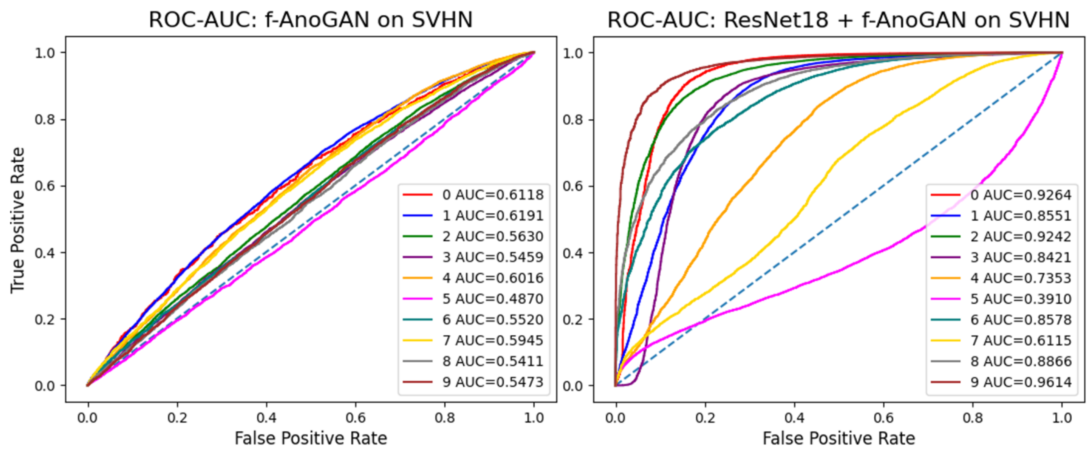

Figure 5.6: Improved Anomaly Detection on SVHN digits through ResNet18+f-AnoGAN compared to original f-AnoGAN architecture

As described in Section 4.4, a score of 1 implies perfect classification performance, that is, correct detection of all anomalies.

With respect to whether a single Anomaly Detection model can be applied to more than a single task (Research Question 3), these results suggest that with the incorporation of a Feature Extractor, the model is applicable to more than one task. As shown in 5.6, the use of this hybrid multi-task model can indeed benefit tasks on other domains by leveraging the transferred knowledge from the USPS and MNIST datasets (as is the case of practically all SVHN digits).

Comparing Figures 5.1 and 5.5, feature extraction does not provide enhanced behaviour for all digits, as some digits, like 4, 7 are only slightly improved. When 5 is trained as the target, f-AnoGAN has a decreased predictive ability than without feature decomposition.

Additionally, a few MNIST digits are not enhanced by the shared knowledge, which might be caused by the excessive noise and heterogeneity found within the SVHN dataset. Indeed, Figure 5.5 shows that some MNIST numbers like 3 and 4 perform worse than without the Feature Extractor. MNIST digit 4 performs only slightly worse, as its original AUC score in Figure 5.1 is barely above the baseline. It is possible that the MNIST digit 4 is particularly challenging for this algorithm. However, it is interesting to note that MNIST number 8 in Figure 5.1 has an average performance of 76%, which increases to 83% with the Feature Extractor, while the MNIST digit 3 has decreased performance. A similar pattern is observed by the digit 3 and 8 of the USPS dataset. It is possible that the noise introduced by the SVHN dataset induces overlap between certain feature concepts, favouring one task in favour of another.

In other domains and applications, the selection of the tasks must be evaluated, as it might not applicable for all tasks (as observed with some digits of the USPS and MNIST datasets).

5.3.3 Analysis on Generalization Ability

This ResNet18 model is pre-trained solely on the target domain (Figure 5.2.2), which enables the model to identify the key features that yield a better model performance and generalization ability. This is demonstrated through its increased Recall score, which experiences high variability across the different digits. The largest overall improvement is experienced on the SVHN dataset, which had the worst scores across all benchmarked algorithms (Table 5.1).

The extraction of the most relevant features per category achieves enhanced generalization ability. However, this affects the interpretation ability of the reconstructed images, as the input images are transformed into pixelated feature maps (Figure 4.6). A drawback of this project's proposed architecture is that f-AnoGAN's interpretability is eliminated due to the use of high-level decomposed features which the human eye cannot easily interpret.

5.3.4 Analysis of Deep and Shallow Feature Extractors

Analysing the effect of deep and shallow feature extractors (RQ2), the ResNet50 model shows that increased feature extractor depth is not particularly beneficial for MNIST or USPS dataset domains on f-AnoGAN. However, the SVHN dataset sees marginal improvement on both its recall and precision scores enhancing the AUC score with respect to the original f-AnoGAN model.

On the other hand, LeNet-5 performance does not, overall, improve performance. It decreases the low generalization ability of f-AnoGAN, but manages to still successfully and consistently identify some few anomalies. The AUC score showcases this model combination is not suitable for complex and highly variable anomalies, and that shallow feature extraction models are not powerful enough to capture the features required for the Anomaly Detection task, especially given the difference between training data fit and test and validation set fit (Figure 5.2).

LeNet-5 outputs a 1-dimensional 144-feature vector, smaller than the 512-feature vectors produced by the ResNet-based models, which can induce important information loss f-AnoGAN requires to successfully learn patterns. The original architecture was strictly modified to imitate the original LeNet-5 model [28], but with the necessary modifications to learn the dataset's requirements as presented in Section 4.3.1.

An additional experiment was run to test whether depth of the feature extractor, and thus dimensionality of the output, affects the results of f-AnoGAN. This experiment further investigates the impact of non-deep feature extraction on Anomaly Detection through dimensionality reduction on the input data using Principal Component Analysis (PCA). To identify the optimal number of features, an analysis of the variance explained by up to 500 Principal Components and its reconstruction error was performed, as shown in 5.7. This resulted in selecting the first 150 principal components of the input data, reducing the dimensionality of the input images and extracting the most relevant features.

The experiment results of the three datasets are reported in Table 5.4, showing that

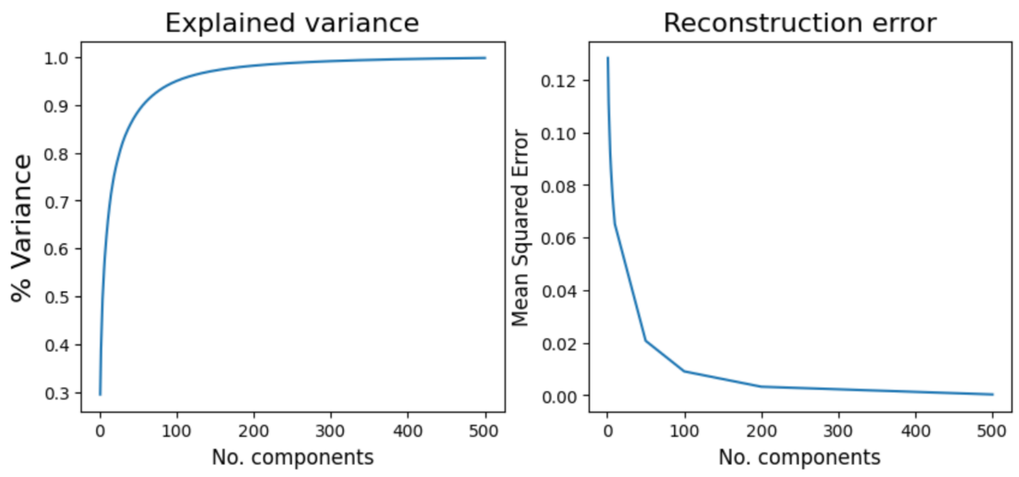

Figure 5.7: Principal Component Decomposition and Analysis

Model	Dataset	AUC	Precision	Recall
PCA + f-AnoGAN	All	74.19 ± 0.68	98.15 ± 0.54	51.07 ± 0.54

Table 5.4: PCA + f-AnoGAN model: Dimensionality Reduction on the input data.

non-deep feature extractors like PCA are strong contenders to deeper alternatives like ResNet18, but also shows that the LeNet-5 is not a suitable extractor for this specific use case.

These results correlate with Erfani et al.'s findings, which state that deeper feature extraction architectures can deliver better generalisation [15]. However, Nahwani et al. proposed further modifications to the LeNet-5 Feature Extractor, like incorporation of a padding and the use of smaller convolution operations, reducing the size of the filters from 5x5 to 3x3[34]. This modification could reduce the number of trainable parameters, which can overfit the data and provide larger, more meaningful feature vectors that f-AnoGAN can benefit from. This sets the groundwork for future work.

Chapter 6

Conclusions

The completion of this project is based two main goals, which were successfully attained. The first goal constitutes model identification and implementation for fair and equal benchmarking of state-of-the-art deep unsupervised Anomaly Detection models on the chosen domain. This was achieved through the implementation and selection of the models f-AnoGAN, Variational Autoencoder, DeepSVDD and Deep Autoencoding Gaussian Mixture Model, respectively. The best overall model with the most competitive performance was found to be f-AnoGAN.

The second, more complex goal required the improvement of the best state-of-the-art model (f-AnoGAN) by leveraging Transfer Learning and Feature Extraction techniques. The completion of this second goal is the main contribution of this project, as a new, multi-tasking hybrid architecture is proposed. It is composed of a deep feature extractor that exploits the shared knowledge gained from semantically-similar datasets to extract the relevant features for the target domain, so an improved generalisation and robustness against outliers is obtained.

This was achieved by implementing and adapting the Dimensionality Reduction techniques and Deep Classification algorithms ResNet, LeNet-5 and PCA. Different depths and complexities of Feature Extractors were tested to investigate their abilities capturing significant features, and the effect on the Anomaly Detection task by using these features as input for f-AnoGAN. These algorithms were transformed into Feature Extractors when trained on a Transductive Transfer Learning objective.

The best Feature Extractor for f-AnoGAN is ResNet18, which performed better than deeper and shallower feature extractors. The combined multi-tasking hybrid model composed of ResNet18 and f-AnoGAN improved the original algorithm's generalization ability and robustness against outliers on all three datasets. The largest improvement was observed on the SVHN dataset, which benefits from the shared feature regularity of the MNIST and USPS datasets. While this architecture manages to improve overall robustness against outliers, the USPS and MNIST datasets observed smaller improvements than those experienced by SVHN.

This proof-of-concept was only evaluated on domains the Feature Extractors had learned as part of the Transductive Transfer Learning objective. It is not known whether the multi-tasking behaviour extends to data the Feature Extractor does not know, that is, whether the multi-tasking objective is maintained without the knowledge of the target domain distribution. This sets the grounds for future work on multi-tasking architectures.

It was also shown that ImageNet-pretrained weights deliver more consistent (less variable) results as opposed to training only on the target domain. This is due to the more complex and complete object representations provided by ImageNet.

Additionally, this project shows that increased depth and complexity of feature decomposition techniques is not correlated with better performance, as PCA provides better performance on the downstream Anomaly Detection task than LeNet-5, but not than the ResNet architectures. Convolutional Feature Extraction works better with residual connections regardless of depth, as all ResNet models tested surpass the LeNet-5 model.

This project thus demonstrates that a hybrid architecture composed of a ResNet18 Feature Extractor trained on the target domain and leveraging features from semantically-related datasets is able to improve the performance and generalization of f-AnoGAN, as well as perform multiple related tasks.

6.1 Further Work

This work has explored the creation of a new multi-tasking hybrid architecture which can enhance generalization ability on state-of-the-art models. The selected semantically-similar domains are selected to develop the cross-domain solution. It must be noted that although similar results can be achieved using other domains, further testing and evaluation is required on realistic domain data to evaluate the cross-domain enrichment on other tasks.

The proposed model is based on a hybrid architecture, which combines both supervised Feature Extraction with unsupervised Anomaly Detection methods. Further work could explore the benefits of a fully-unsupervised feature extractor using alternative Deep Classification networks like Deep Belief Networks [23] or through the additional modification of the ResNet architectures through the development of unsupervised loss functions.

As presented in the Experiments section, Nathwani et al.'s proposed LeNet-5 model modifications could also be attempted to investigate the effects of richer feature vector [34]. Altenatively, further work could also investigate whether the findings align with Erfani et al.'s affirmation that removal of redundancy in feature vectors enhances performance, by using very low-dimensional feature representation [15].

Bibliography

[1] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

[2] Charu C. Aggarwal. *An Introduction to Outlier Analysis*, pages 1–34. Springer International Publishing, Cham, 2017.

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

[4] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*, pages 622–637. Springer, 2019.

[5] Jinwon An and Sungzoon Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special lecture on IE*, 2(1):1–18, 2015.

[6] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. *Advances in neural information processing systems*, 19, 2006.

[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[8] BJ Bakker and TM Heskes. Task clustering and gating for bayesian multitask learning. 2003.

[9] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[10] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9592–9600, 2019.

[11] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. Anomaly detection using one-class neural networks. *CoRR*, abs/1802.06360, 2018.

[12] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41, 07 2009.

[13] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A. Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018.

[14] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240, 2006.

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

[16] Theodoros Evgeniou, Charles A Micchelli, Massimiliano Pontil, and John Shawe-Taylor. Learning multiple tasks with kernel methods. *Journal of machine learning research*, 6(4), 2005.

[17] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[18] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[19] Kasthurirangan Gopalakrishnan, Siddhartha K Khaitan, Alok Choudhary, and Ankit Agrawal. Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection. *Construction and building materials*, 157:322–330, 2017.

[20] Songqiao Han, Xiyang Hu, Hailiang Huang, Mingqi Jiang, and Yue Zhao. Adbench: Anomaly detection benchmark, 2022.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[22] Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. *Pattern recognition letters*, 24(9-10):1641–1650, 2003.

[23] Geoffrey E Hinton. Deep belief networks. *Scholarpedia*, 4(5):5947, 2009.

[24] J.J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.

[25] Simon Kornblith, Jonathon Shlens, and Quoc V. Le. Do better imagenet models transfer better? In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2656–2666, 2019.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[27] Kaisar Kushibar, Mostafa Salem, Sergi Valverde, Àlex Rovira, Joaquim Salvi, Arnau Oliver, and Xavier Lladó. Transductive transfer learning for domain adaptation in brain magnetic resonance image segmentation. *Frontiers in Neuroscience*, 15:608808, 2021.

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

[29] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

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

[31] Francesco Lupo. *Variational Autoencoder for unsupervised anomaly detection*. PhD thesis, Politecnico di Torino, 2019.

[32] Maryam M Najafabadi, Flavio Villanustre, Taghi M Khoshgoftaar, Naeem Seliya, Randall Wald, and Edin Muharemagic. Deep learning applications and challenges in big data analytics. *Journal of big data*, 2(1):1–21, 2015.

[33] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? *arXiv preprint arXiv:1810.09136*, 2018.

[34] Chetan L Nathwani, Jamie J Wilkinson, William Brownscombe, and Cédric M John. Mineral texture classification using deep convolutional neural networks: An application to zircons from porphyry copper deposits. *Journal of Geophysical Research: Solid Earth*, 128(2):e2022JB025933, 2023.

[35] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[36] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

[37] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. *Advances in neural information processing systems*, 29, 2016.

[38] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[39] Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795, 2021.

[40] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4393–4402. PMLR, 10–15 Jul 2018.

[41] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2015.

[42] 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.

[43] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[44] 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.

[45] 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 *Information Processing in Medical Imaging: 25th International Conference, IPMI 2017, Boone, NC, USA, June 25-30, 2017, Proceedings*, pages 146–157. Springer, 2017.

[46] Bernhard Scholkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, John Platt, et al. Support vector method for novelty detection. *Advances in neural information processing systems*, 12(3):582–588, 2000.

[47] VAE. Variational autoencoder in tensorflow. `https://learnopencv.com/variational-autoencoder-in-tensorflow/`. Accessed: 2023-03-01.

[48] Jie Yang, Ruijie Xu, Zhiquan Qi, and Yong Shi. Visual anomaly detection for images: A systematic survey. *Procedia Computer Science*, 199:471–478, 2022. The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 and 2021): Developing Global Digital Economy after COVID-19.

[49] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? 2014.

[50] Yu Zhang and Qiang Yang. An overview of multi-task learning. *National Science Review*, 5(1):30–43, 2018.

[51] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsuper-

vised anomaly detection. In *International conference on learning representations*, 2018.

Appendix A

Supporting Information

A.1 Experiments: model settings and parameters

This section provides details for reproducibility of the Benchmarking Experiments results. At training time, the network's weights are updated on each batch via backpropagation with the objective of minimizing the corresponding loss functions pertaining each algorithm.

All algorithms were trained on 3 times 20, 100, 150 and 200 epochs, using random seeds for each iteration. Image size of 28x28 and batch size of 64 were used, with pre-processing as described in Chapter 3. PyOD's environment with its corresponding dependency versions were used.

- The f-AnoGAN algorithm was trained using the Adam ooptimizer was used with a learning rate of $1e$-4, and default learning rate decay values found in the code files. A latent space dimensionality of 100 was also used.

  The Feature Extractor + f-AnoGAN experiments were run on a seed value of 2.

- The VAE model used a size 100 for the dimensionality of the latent space, gamma value of 0.8 and capacity of 0.2 for all its runs. MSE loss function with Adam optimization and Sigmoid activation function, with all the corresponding default parameters found in the code. The model was run on the ADBench environment.

- The DeepSVDD approach used an Autoencoder, following Ruff's own implementation and PyOD's default setting, with a sigmoid activation function, Adam optimizer ReLU activation function. Default parameters as found in the code were used.

- The DAGMM model was trained using Adam optimization with a default latent dimension of 1, the value of 10 for the components was found to be the most successful, and the lambda energy value 0.1 and lambda covariance to penalize the small values on the diagonal of the covariance matrix has a very small value of 0.005.

A.2 Additional Feature Extractor + f-AnoGAN results

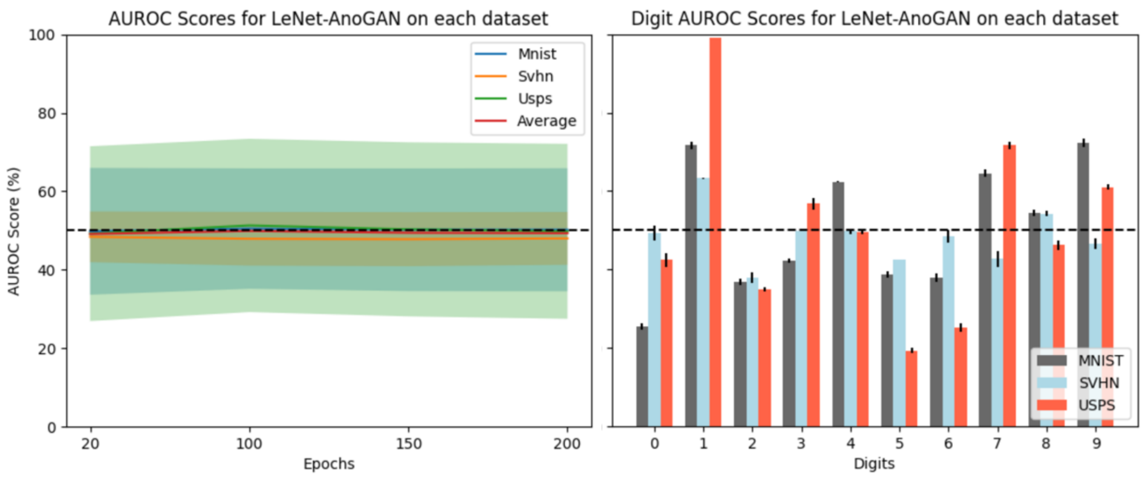

Figure A.1: **LEFT**: AUROC Scores of LeNet-5 Feature Extractor with f-AnoGAN base model, for each dataset across epochs for all digits. **RIGHT**: AUROC Scores of f-AnoGAN when trained on each target digit, for all datasets using LeNet-5 as Feature Extractor

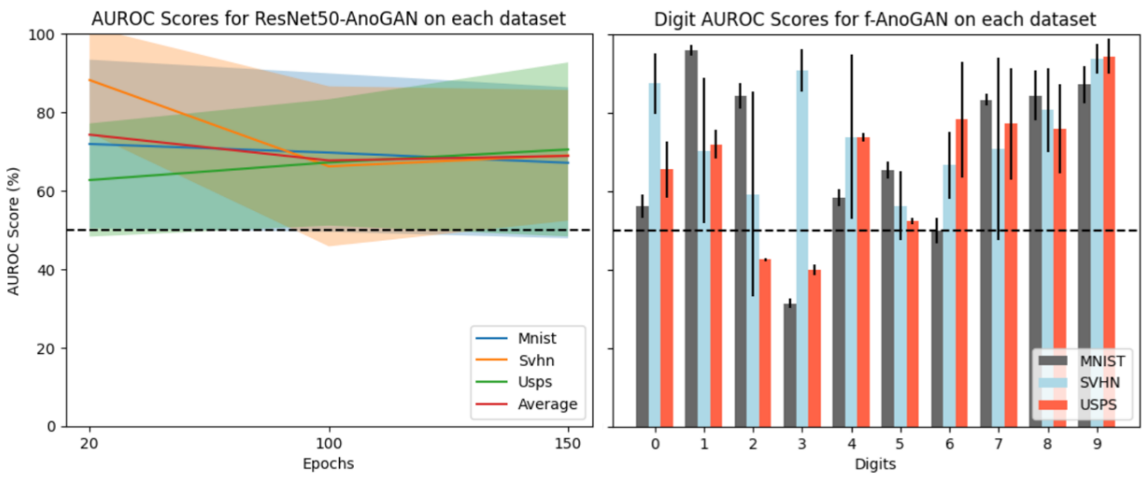

Figure A.2: **LEFT**: AUROC (AUC) Scores of ResNet50 + f-AnoGAN, for each dataset across epochs for all digits. **RIGHT**: AUROC Scores of ResNet50 + f-AnoGAN when trained on each target digit, for all datasets