UNIVERSITY
OF UDINE

Department of
Mathematics, Computer Science and Physics

PH.D. THESIS IN
COMPUTER SCIENCE MATHEMATICS AND PHYSICS 313

Deep Neural Networks for Image Anomaly Detection: Application in Real World Industrial Scenarios

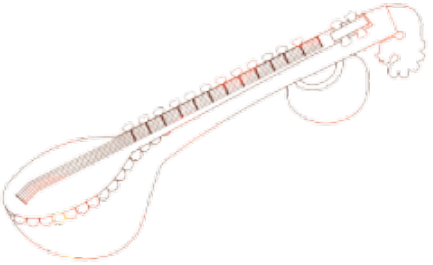
CANDIDATE
Pankaj Mishra

SUPERVISOR
Prof. Gian Luca Foresti

Cycle XXXIV — A.Y. 2020-2021

Institute Contacts
Dipartimento di Scienze Matematiche, Informatiche e Fisiche
Università degli Studi di Udine
Via delle Scienze, 206
33100 Udine — Italia
+39 0432 558400
https://www.dmif.uniud.it/

Author's Contacts
Via Monte Grapa, 20
Udine — Italy
+39 3201434039
mishra.pankaj@spes.uniud.it



Dedicated to the almighty and my beautiful family for keeping their faith in me...!!

Abstract

Deep neural network is the new norm in the present era, where it's being used in almost all the evolving fields, so is the field of anomaly detection. With the modern IT infrastructure industries are in constant quest to search for new algorithms to analyze the data in order to be more autonomous, agile, efficient, and cost-effective. *Anomaly detection* is one such task, which industries want to automate as it finds its application in various fields like banking, traffic management, manufacturing, online fraud detection, anomalous behavior, etc.

In this dissertation, we explored various novel approaches to solve the image anomaly detection problem. We proposed ways for anomaly detection tasks, keeping real industrial problems in mind like data scarcity, imbalanced data, limited resources, etc. The dissertation proposes novel models like adapted *capsnet, stacked capsule auto-encoders, Pyramidal Image Anomaly Detector* (PIADE), and *Vision Transformer for Image Anomaly Detection and Localization* (VT-ADL) for Deep Anomaly Detection (DAD) task. The methods can be trained in both supervised and semi-supervised ways. And VT-ADL is capable of global image classification and localization, without any need for pixel-precise ground truth data. We tested our methods on various academically used datasets like *MNIST, CIFAR, COIL, FMNIST*, and real industrial datasets like *MVTec* and *BTAD*. The proposed methods performed at par or better than the state-of-the-art methods.

Declaration

I declare that the dissertation has been composed by myself and that the work has not to be submitted for any other degree or professional qualification. I confirm that the work submitted is my own, except where work that has formed part of jointly-authored publications have been included. My contribution and those of the other authors to this work have been explicitly referenced. I confirm that appropriate credit has been given within this thesis where reference has been made to the work of others.

Name: Pankaj Mishra

Date: October 28, 2021

List of Publications

- ***Journal Publications***

  - **Mishra Pankaj**, Claudio Piciarelli, and Gian Luca Foresti. *"A neural network for image anomaly detection with deep pyramidal representations and dynamic routing."* International Journal of Neural Systems 30.10 (2020): 2050060.
  - Piciarelli Claudio, **Pankaj Mishra**, and Gian Luca Foresti. *"Supervised anomaly detection with highly imbalanced datasets using capsule networks."* International Journal of Pattern Recognition and Artificial Intelligence (2021): 2152010.
  - **Mishra Pankaj**, Riccardo Verk, Claudio Piciarelli, Gian Luca Foresti. *"Comparative study of segmentation nets and patch-based methods for anomaly detection in industrial scenarios."* Engineering Applications of Artificial Intelligence, The International Journal of Intelligent Real-Time Automation (2021), (*submitted*)
  - Nardin Axel, **Mishra Pankaj**, Gian Luca Foresti, Claudio Piciarelli. *"MeTAL - Masked Transformer for Anomaly Localization"* IEEE Transactions on Industrial Informatics (2021). (*submitted*)

- ***Conference Publications***

– **Mishra Pankaj**, Verk Riccardo, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. *"VT-ADL: A Vision Transformer Network for Image Anomaly Detection and Localization"* 30th IEEE/IES International Symposium on Industrial Electronics (ISIE) Kyoto, Japan, June 20-23, 2021

– **Mishra Pankaj**, Claudio Piciarelli, and Gian Luca Foresti. *"Image anomaly detection by aggregating deep pyramidal representations."* International Conference on Pattern Recognition. Springer, Cham, 2021.

– Piciarelli, Claudio, **Pankaj Mishra**, and Gian Luca Foresti. *"Image anomaly detection with capsule networks and imbalanced datasets."* International Conference on Image Analysis and Processing. Springer, Cham, 2019.

Abbreviations

CNN : Convolutional Neural Network

GB: Giga bytes

TB: Tera bytes

IOT: Internet of Things

VIS: Visual Inspection Systms

NLP: Natural Language Processing

CL: Continuous Learning

DL: Deep Learning

AI: Artificial Intelligence

ML: Machine Learning

FSL: Few Shot Learning

DNN: Deep Neural Networks

DAD: Deep Anomaly Detection

Acknowledgements

"*A journey of a thousand miles begins with a single step*", is a common Chinese proverb, taken from chapter 64 of ancient Chinese classic text; *Tao Te Ching* attributed to *Lao Tzu*. The proverb is instilled in me and when I look back to my journey in Europe for the past 5 years, I see a whole spectrum of human experiences.

I don't remember when exactly I developed a passion for research, but I am very sure that it started from a very tender age. Looking hindsight I can see that the rapidly changing world and at the same time the pace of changing technology has played a key role in many of my decisions. So first of all I want to express my gratitude to all communities around the world who are striving to make this world a better place for the coming generation.

Swami Vivekananda once said, "*Comfort is no test of truth. Truth is often far from being comfortable*". His words resonate inside me every single day. I would not lie in saying that the Ph.D. was an easy ride. It has been tough, even exhausting at times, but highly rewarding, especially from a human perspective. There were times when I was flying high and there were days when I seemed doomed. But as when I felt doomed, and I felt of going back to my cocoon, the little larva inside of me said just one more step..!!

It would be unfair, indeed, to present this dissertation as the result of my only doing, being it rather the culmination of extensive collaborations and discussions of many talented people who helped me at many different levels throughout my doctoral

path.

In primis, I would like to thank my colleagues and friends at the department of computer science physics and mathematics at Udine: *Rao Mohammad Umer, Asad Munir, Dr. Vaibhav Bansal, Riccardo Verk, and Matteo Dunnhofer* who helped me with their experience and the pleasant company almost on a daily basis.

A special thanks go to *Bon Fabrizio* and *Giuseppe Piscopo* for being outstanding support and awesome flatmates during my initial days in Udine. Nonetheless, I also want to thank all those new friends like *Dr. Vaibhav Bansal, Dr. Anitha Rajendran, Dr. Nisha Sharma, Dr. Manish Kumar, Showmeya Mallavarapu* who became part of this journey in both formal and informal ways. Without you guys, this journey would have lacked those vibrant colors, whom I cherish the most.

All heartfelt thanks and respect to all those who touched my life outside the university scope. My family, *Radhey Shyam Mishra, Lili Mishra, Deepak Mishra, Piyush Mishra*, and *Anjali Mishra* who have never constrained me and have always been present in times of need; my historical friends *Megha Marwari, Akshay Malik*, and *Narendra Singh* for giving me so much support and comfort also outside the university space.

Finally, words cannot describe how grateful I am to my supervisor *Prof. Gian Luca Foresti*, and *Prof. Claudio Piciarelli*, who guided me during this doctoral journey by answering thousands of e-mails, helping me with incredibly valuable insights, and providing me with a huge amount of time to discuss and develop my research ideas. His expertise and almost paternal advice, his passion, and his dedication have thought me how beautiful and motivating research can be. Additionally, I want to express gratitude to *Prof. Christian Micheloni* and *Prof. Niki Martinel* for time to time taking part in healthy discussions and providing support when and as needed. I would also like to express my heartiest gratitude to the entire beanTech company for sponsoring my Ph.D. research, especially *Daniele Fornasier*, who took active participation in various technical topics and gave his expert insights on them. Apart from being an acquaintance for my

research work Daniele also showed me his love for Indian culture and cuisine which culminated into the various interesting discussions which tried to shape my perspective on life and technology.

Thanks to the thinkers and all the other friends I could not mention in this brief acknowledgment section, who, during my Ph.D., significantly shaped my professional and personal life and to the many people I crossed paths with, who, simply touching it, made it a unique and wonderful journey

Contents

List of Figures

List of Tables

1

Introduction

"Every industrial revolution brings along a learning revolution."

– *Alexander De Croo*

It has been 10 years since Industry 4.0 [17] was first introduced as a concept by a German consortium to improve the country's competitiveness in the manufacturing industry [39]. The fourth industrial revolution was defined as the automation of traditional manufacturing and industrial practices and has the aim to take manufacturing processes to a new level. But the barriers proved daunting and what was touted as a revolution turned into a slow evolution [104]. Artificial Intelligence (AI) is proving to be a game-changer for production and manufacturing, enabling intelligent factory automation that is faster and cheaper than ever before. In addition, many industry experts are using AI for quality control activities. But to achieve industry-grade real-life AI solutions is tough because still, AI solutions are problem-specific, data hungry, computationally demanding and work on a single modality(means, only image data, or sensory signals, or sound, or tabular data). Hence, there is a constant strive among

the academic and scientific community and industry as well to develop more robust and fewer data-hungry and easy-use systems for real-life applications.

According to medical terms in cognitive psychology [9], *"novelty"* is in a total of the *remembered* events plus some degree of *surprisal* aroused from an event observation. And the surprising part can be mathematically modeled with a low probability to occur in an expected model.

Anomaly detection is referred to as the process of identifying novel samples that exhibit significantly different traits with an accepted and pre-defined model of normality. In real-life scenarios, like Visual Inspection Systems (VIS), the novel sample can show an unseen considerable surprise, which at that point is not following the pre-defined normality, and labeling of novel examples are not possible. If we follow the literature such event were solved by approximating the ideal shape of the boundary separating normal and novel samples by modeling the intrinsic features of the former. Due to the increasing demands, ease of application, cost-saving, and security enhancement, anomaly detection is seeing broad domains of applications and has been an active research area for several decades, with earlier attempts dating as far as the 1960s [40]. However, from the practical point of view anomalies can be seen in one of three types (See Fig:1.1) [21]:

- **Point Anomalies**: It represents anomalies that happen randomly and has no particular pattern or interpretations. Most of the old literary work is done to tackle this kind of problem.

- **Contextual Anomalies**: It is also known as *Conditional Anomaly* which identifies anomalies based on some contextual and behavioral information. It usually uses time and space as the contextual feature while it uses a specific pattern or certain behavior as a behavioral feature.

- **Collective or Group Anomalies**: A gathered point of data is classified in this group if the individual point in this collection seems normal while when observed

in a group exhibits unusual characteristics.

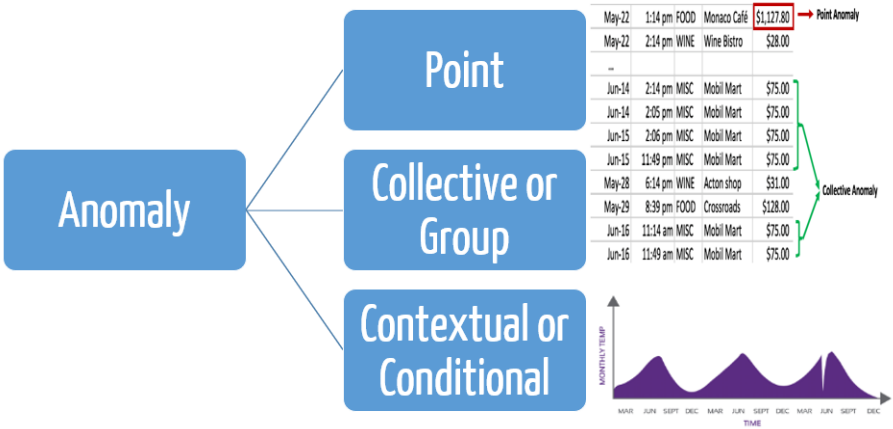


Figure 1.1: Classical classification of Anomalies

Anomaly detection is an important problem that has been studied in different research areas and application domains. Many anomaly detection techniques have been developed specifically for certain application domains, while others are more general. In computer vision, an anomaly is any image or image portion which exhibits significant variation from the pre-defined characteristics of normality. Anomaly Detection is thus the task of identifying these novel samples in supervised or semi-supervised or unsupervised ways. A system that can perform this task in an intelligent way is hugely in demand, as its applications range from video surveillance [85] to defect segmentation [84, 25, 70], inspection [84], quality control [72], medical imagining [64], financial transactions [127] etc. As it can be seen from the examples, anomaly detection is particularly significant in the industrial field, where it can be used to automatically identify defective products.

In recent years deep learning has shown tremendous capabilities in automating various mundane industrial tasks by learning deep expressive representations of complex data (high-dimensional data), spatial data, temporal data, tabular data, images and

videos, and graph data, pushing the boundaries of various research areas, and anomaly detection is one such field. Deep learning for anomaly detection is also known as *deep anomaly detection*, which aims for learning the feature representation or an anomaly score via a deep neural network. Initial work on deep learning (and Machine Learning) tries to classify anomaly detection in two main tasks (Fig: 1.2):

- **Novelty Detection**: A mechanism by which an intelligent organism/system is able to identify an incoming sensory pattern as being *until now unknown*. This has huge application in bio-medical fields [35], manufacturing fields [120], banks and online transactions [7], trading [117], etc. In this approach, the training data is not polluted by the outliers and we are interested in detecting whether a new observation is an outlier. In this context, an outlier is called a *Novelty*.

- **Outlier Detection**: An outlier is an observation that diverges from an overall pattern on a sample. And the task of finding that is called *Outlier Detection*. In this approach, the training data contains outliers, which are defined as the observations that are far from the others. Thus these methods try to fit a region where the training data is the most concentrated ignoring the deviant observations.

But in terms of learning these two methods differs; where Novelty detection is a Semi-supervised learning in contrast to Outlier detection as Unsupervised learning (see Fig 1.3).

Learning the local features and keeping the positional information in an unsupervised way is a novel and very challenging task in the computer vision domain. The most informative part of any image dataset is the part with the highest variance. Such regions can be located in a very small as well as large local area in an industrial image. Hence, extracting those pixel-precise regions, in an unsupervised way and at the same time preserving the spatial information is often desirable, as it helps in automating various industrial scenarios. Most of the classical machine learning techniques try to learn this

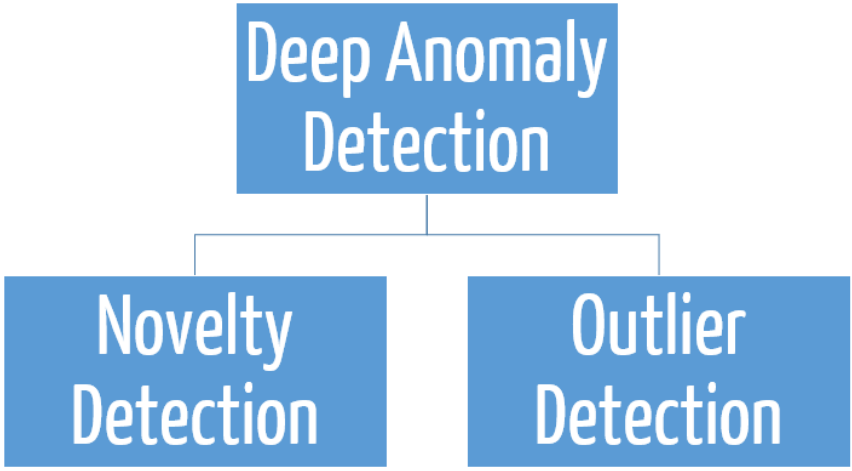


Figure 1.2: Early attempt to classify deep anomaly detection tasks

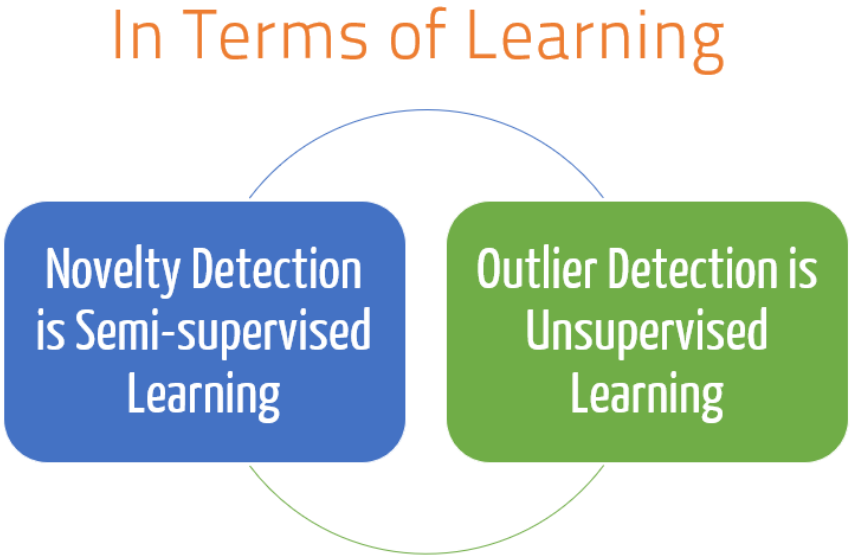


Figure 1.3: Learning the difference between Novelty and Outlier Detection approaches

variance (also termed as *Entropy* [38]) to find the *causal features* [38] in the dataset. With the rise in modern IT infrastructure, there is a boom in data acquisition, which subsequently requires better and efficient algorithms to be processed. And this leads to

our motivation for this research work.

1.1 Motivation

Quality inspection is often carried out by workers on the production line and manufacturers are experiencing serious labor shortages and inconsistent quality assessments by human workers [104]. Experts have estimated that the *"Cost Of Poor Quality"* (or COPQ) [17] can be as high as 30 percent of gross sales for manufacturing and service companies, costing them millions of dollars each year, so there is high motivation to improve quality inspection and prevent product defects earlier in the process [89].

With decades-long research and earlier attempts, industries are seeing themselves on the verge of rapidly evolving interconnected systems, where new technologies have pressing demands to keep the industries competitive and innovative. Following these demands recent efforts have been made to improve the anomaly detection task using deep learning. Most of the works try to learn the manifold of a single class representing normal data [124], using an encoding-decoding scheme, and their output is a classification of the input image as either normal or anomaly, while fewer works deal with the task to segment the local anomalous region in an image [12]. Majorly, the methods either use a reconstruction-based approach or learn the distribution of the latent features extracted by a pre-trained network or trained in an end-to-end fashion [78]. Availability of training data is also a big challenge, as most of the industries never share a lot of data because of privacy issues and even if they share, the data tends to be highly imbalanced.

Since anomaly detection task has huge industrial application, many industries want to develop there own specialized methods which suits their requirements of anomaly classification is a complex working domain. The major and prime industrial challenge is the "lack of data". Many industrial facilities still don't have proper infrastructure for ideal data collection facilities. Even if they have; they have a huge data imbalance, as

data for the normal product is easy to produce, while very few images of the abnormalities are available as they add-up to the COPQ. Quality of data is also important and many times industries find it very difficult to arrange that, as they may have huge piles of messy data.

Apart from the above-mentioned challenges, it's also costly to arrange customized industrial data because this may affect the entire production schedule. And even if we get the data, its annotation is costly(both in terms of time and money). Table 1.1 shows the major challenges with industries and deep anomaly methods.

Vanilla Neural Network	**Industrial Challenges**
All Vanilla Deep Neural Networks are data hungry	Scarcity of data or standard dataset because of privacy issues
Most of the classical deep anomaly methods need annotated data	Annotation is costly and time-consuming
Highly imbalanced datasets induces biased learning or overfitting	Very high-imbalanced datasets are very common in industrial setups

Table 1.1: Challenges with industries and Vanilla Neural networks

Hence, the industrial challenges and the benefits of recent advances in the deep learning field being our major motivation for our research work, which includes -

- **Low Recall Rate of Anomaly Detection**: Though with the decades of research the classical research still had high *false positive* [88] on real world datasets [18, 79]. Hence, our motivation is to develop deep anomaly techniques to reduce the false positive and enhance detection recall rates (see chapter 5.2).

- **High-Dimensional Anomaly Detection**: Old classical approaches were more limited to low dimensional space, where a separating plane between the normal features and novel data point is fitted [51, 58, 83] or feature engineering/selection is done to separate the normal data points with the novelty [76, 77, 6]. While at higher dimensions usually anomalies have intricate interaction and it's tough to identify them. Higher-dimensional space (e.g. images or videos) poses further

challenges as they are not spatial invariant, hence, the methods need to preserve the information for a specific orientation. Hence, our motivation is to develop methods which can do real industrial class anomaly classification and location using high-dimensional data (images).

- **Imbalanced-Data Learning**: Cost, difficulty, and privacy issues related to the collection of large-scale labeled data, often makes *fully supervised learning* impractical to implement. However, in recent times trend has been changed to develop more *semi-supervised or unsupervised learning*, that doesn't (or minimum) require labels for training [86, 67, 68, 70]. Hence, the biggest motivation our research work is to learn expressive normality/abnormality representation with the novel deep networks with limited(scarce) data which can generalize to the novel anomalies.

- **Resilient Anomaly Detection and Localization**: With deep networks resilient feature learning is expected, as deep networks are susceptible to changing luminosity. Conventionally the anomaly localization part is done through various segmentation networks like Unet [95], Unet++ [131], UNet2 [48], SegNet [8]. But these segmentation networks need lots of labeled data (see chapter 7) and are also hard to train and slower in inference. Additionally, our quest is to develop novel deep network that can do global image-level anomaly classification as well as anomaly localization. Indeed, while doing this the network should be fast enough to be deployed for industrial uses.

1.2 Problem Definition

This thesis outlines novel **deep anomaly detection** (DAD) methods, whose mathematically problem definition can be can set in the pretext of some real-world industrial acquired dataset.

Hence, given a dataset $\mathcal{X} = \{I_1, I_2, I_3, ..., I_N\}$ with $I_i \in \mathbb{R}^D$, let $\zeta \in \mathbb{R}^K (K \ll N)$ be the feature representation space, then deep anomaly classification methods aims to learn a feature mapping function $\phi(\cdot) : \mathcal{X} \mapsto \zeta$. On the other hand, deep anomaly scoring methods, learns a function $\tau(\cdot) : \mathcal{X} \mapsto \mathbb{R}$, in a way that anomalies can be easily pointed out from the normal instances in the space yielded by the functions ϕ or τ. In the classical approaches these ϕ or τ are closed form convex functions, which usually shows poor performance with high dimensional data. But here ϕ and τ are trainable neural networks with $H \in \mathbb{N}$ hidden layers and their weight matrices $\Theta = \{M^1, M^2, M^3, ..., M^H\}$. In case of classification $\phi(\cdot)$ produces probability or class prediction, while $\tau(\cdot)$ directly infers the anomaly scores for a novel instance.

1.3 Thesis Outline

The dissertation is organized as follows. Chapter 1 introduces the background on deep anomaly detection, the motivation behind it, and problem definition. Chapter 2 presents the related work done previously in the field of deep anomaly detection. Chapter 3 presents our novel approach for image-level anomaly classification using an adapted capsule network for the high-imbalanced dataset. Chapter 4 and Chapter 5 show our novel semi-supervised approach for the image novel anomaly classification methods where we used "Stacked Auto Encoders Using Pyramidal Features for Global Image Anomaly Classification" and "Pyramidal Image Anomaly Detector (PIADE)" respectively. Chapter 6 presents the novel use of Vision Transformer and Gaussian mixture model for anomaly classification and localization (VT-ADL). Chapter 7 present an important comparative study between the newly developed patch-based models like VT-ADL in contrast with other segmentation networks from an industrial point of view. Following this Chapter 8 discusses the future scope of work and finally, Chapter 9 shows the conclusion of this dissertation.

2

Related Work

"Our intelligence is what makes us human, and AI is an extension of that quality"

– *Yann LeCun*

From the previous chapter 1 it can concluded that anomaly detection methods are an active research field, due to its vivid applications. Also, traditionally it has been studied widely using classical machine learning approaches. A recent survey by Chandola et al. [22] gives an excellent overview of the topic, and all possible non-deep approaches. From a deep learning point of view Kiran et al. [53] has published a survey showcasing all the recent deep anomaly detection approaches in images and videos. Another good survey by Pang Guansong et al. [78], excellently tried to discuss the advantages and disadvantages of the various deep learning approaches used in the recent times.

From the excerpts of literature we can find that the deep anomaly detection can be classified based on learning (training) methods and conceptual paradigms. Both of them are discussed below respectively.

2.1 Categorization based on Learning Methods

Learning methods in Deep Neural Networks (DNN) refers to the technique used for training. As deep networks are data hungry models, but they demand a well prepared dataset. Based on these data requirements for training, DNN are classified as (in our case for deep anomaly detection task):

- **Supervised Anomaly Detection** methods are prominently used to handle imbalanced datasets. When the dataset is imbalanced, the previous works dominantly use generic techniques such under-sampling the dominant class or oversampling the smaller class either by data duplication or by synthetic generation of new data. In both cases the idea is to use a pre-processing step to make the dataset balanced before applying any classification algorithm[16]. These methods do not always lead to good results, since oversampling produces an over-fitted model or the model doesn't learn proper generalization from the training dataset. Also, when highly imbalanced dataset is used for deep learning models, the model learns almost nothing about the anomalies and gets over-fitted on the normal classes. Although the supervised methods have improved their performance over the time, but not as popular as the semi-supervised or unsupervised methods, owing to the lack of labels in the training data. Moreover, as discussed earlier the sub-optimal performance of the supervised models due to class imbalance further made it more infamous.

- **Semi-Supervised Anomaly Detection** approach, early work has adopted the deep learning techniques such as Deep Belief Networks[124] for medical image analysis or Restricted Boltzmann Machines[37] for network traffic analysis. The most recent approach is to use the traditional methods using the deep learning approach, one of such work is done by Ruff et al.[97] propose the Deep Support Vector Data Description method, in which a deep neural network is trained under

the same constraints adopted by one-class Support Vector Machines. The majority of current deep learning methods uses autoencoders or generative models. In many practical applications, e.g. in medical [35] and industrial fields [120], the label for the normal class is quite easy to obtain, while the label for anomalies is rarely available or is very scarce. Hence, using the autoencoders and training them only on the normal data in a semi-supervised way is the most commonly opted method [67, 68]. As normal samples are sufficiently available, autoencoders are trained for the normal classes while training. The trained autoencoder will produce low reconstruction errors for the normal instances, while it will produce high reconstructions error for the anomalies[71][112].

- **Unsupervised Anomaly Detection** methods for the anomaly detection are solely based on the intrinsic features of the data (outlier detection). Various deep learning methods have shown to outperform the traditional machine learning methods like Principal Component Analysis (PCA) [123] and Support Vector Machine (SVM) [113]. While there is a more recent approach to use the Generative Adversarial Networks(GAN) [28]. Such generative networks are based on the two competing networks: a generator, which generates a new unseen data resembling to the training datasets and a discriminator, which tries to discriminate between the original dataset and the generated dataset. For the anomaly detection these GAN networks are trained on the normal class (because of the easy availability and abundance of normal class data) much like autoencoders. At the time of testing the generator is inverted, which provides the comparison between the latent space representation of the normal and the anomalous data[29, 102, 2].

2.2 Categorization based on Conceptual Paradigms

In literature, we can also find different classification based on adopting different learning and scoring strategy of the networks [70, 11, 55, 93] like: a) Reconstruction based approach, b) Latent space learning based approach, c) Generative network based approach. However, there is no strict division between each of the approaches, as many of these approaches overlaps with one or more methods used. Hence, the three approaches shown in Figure 2.1 can serve as the reference for any of the other classification methods present in contemporary literature.

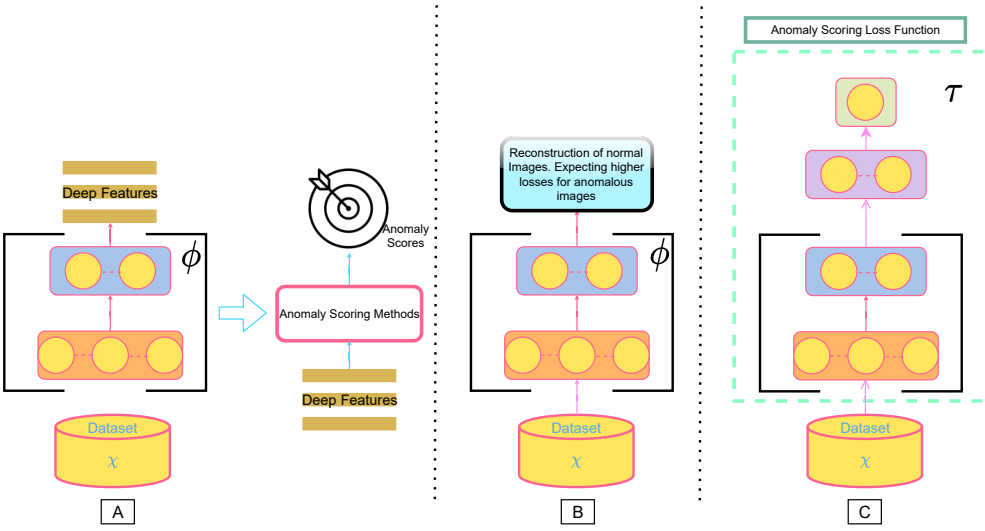


Figure 2.1: Categorization of three main conceptual frameworks of Deep Anomaly Detection Approaches. A) Pre-trained networks are used as feature extractors and then an anomaly scoring method is employed to score the images. B) Normality manifold is learned via deep network using reconstruction-based methods. C) Anomaly scores are learned in an end-to-end fashion.

To have a thorough understanding, apart from the learning method (see 2.1), the approaches are categorised based on three conceptual paradigms [78]:

- *Feature Extraction Using Pre-Trained Deep Networks*

- *Learning Feature Representation of Normality (Normality Manifold learning)*
- *End-to-end Anomaly Score Learning.*

2.2.1 *Feature Extraction Using Pre-Trained Deep Networks*

All the methods [10, 38] which leverages a pre-trained state-of-the-art network to extract the low-dimensional feature representations from high-dimensional data (images) for anomaly detection can be classified in this category. The step of feature extraction and anomaly scoring are fully disjoint and independent from each other. Hence, deep learning models are only used for leveraging their dimensional reduction property while keeping the most important features intact. Formally, the method can be written as:

$$\mathcal{Z} = \Phi(X; \Theta) \tag{2.1}$$

where $\Phi : \mathbf{X} \mapsto \mathcal{Z}$ is a pre-trained deep neural network, which maps the input image $\mathbf{X} \in \mathbb{R}^D, \mathcal{Z} \in \mathbb{R}^K$ and $D \gg K$. While a separate anomaly scoring function f is employed, which has no connection to the feature extraction step. It's applied on the $\mathcal{Z}$ to calculate the anomaly score.

The underlying assumptions in all such work [67] is that the deep features extracted preserves the discriminative information, while reducing the dimension, which helps to separate anomalies from the normal instances. Most popular networks used in such methods are AlexNet[57], VGG[109] and ResNet[41]. The advantages of these methods is large number of off-the-shelf state-of-the-art networks, trained on large imagenet or similar datasets, and are easily available. Deep networks provide much powerful dimensionality reduction using complex approximations, filtered through it's deep layers, than most of the popular linear and non-linear methods [19, 103]. In addition, it's very easy to implement these methods, as most of the deep learning platforms are facilitating these trained networks and can be adapted in just few lines of codes. Disadvantages

of this method is that the fully decoupled method of feature extraction and anomaly scoring often led to sub-optimal scoring. In addition to this, there is minimal control over the learned manifolds of the pre-trained models.

2.2.2 *Learning Feature Representation of Normality (Normality Manifold learning)*

In this approach , the steps of feature extraction and anomaly scoring are coupled/overlapped in some ways, rather than fully disjoint in the previous approach. The methods learn the representation of normal images by optimizing a generic feature learning objective function; that is primarily designed to replicate the input image data. But the learned representation is still used for the anomaly detection task, since the learned network captures the key underlying data regularities. Here a deep neural network is trained, either in semi-supervised or unsupervised fashion to learn the manifold of the normal data in it's latent space. While this latent space is used to reconstruct the input data, in the process by minimising some likelihood loss function, mainly reconstruction losses. Formally it can be written as -

$$\{\Theta^*, \mathbf{W}^*\} = \underset{\theta, \mathbf{W}}{\operatorname{argmin}} \sum_{x \in \mathcal{X}} \ell(\psi(\Phi(X; \Theta); \mathbf{W})), \tag{2.2}$$

$$S_x = f(x, \Phi\Theta^*, \psi\mathbf{W}^*), \tag{2.3}$$

where Φ maps the input image data to the latent representation space $\mathcal{Z}$, ψ parameterized with $\mathbf{W}$ is the learning task that takes the latent representation $\mathcal{Z}$ and enforces the the learning of the normal data manifold, ℓ is a loss function used in the underlying approach, and f is scoring function that is based on Φ and ψ to calculate the anomaly score.

Normality manifold learning includes methods driven by various perspectives, including reconstruction based method, generative models, latent space learning, pyramidal features extraction in both semi-supervised and supervised learning fashion. In most of the approaches they used autoencoders [67, 68, 46, 42]. This primarily uses encoding decoding scheme on which the normal data can be well reconstructed. The underlying assumption is that the normal class can be better reconstructed in contrast to the anomalous images, leading to higher losses for the later class. The advantages of such methods are they are very straightforward to any type of data and easy to implement. Additionally, various different type of AE's like Variational AE[5], Contractive AE[94], Denoising AE[118] can be used along with complex loss functions, like Structural Similarity Index (SSIM)[122] and perception loss [68]. The generic approaches tried resulted good in various real life scenarios, but it suffers huge when the input images are disorientated or with change in illumination.

To counter such shortcoming of the AE based reconstruction approaches, Generative Adversarial Network (GAN) based anomaly detection methods [2, 101] emerges quickly, as it aims to learn the latent feature space of the Generative Network. The latent space tries to capture the normality underlying the given data. Subsequently, the residual between the real instance and the generated instance are then defined as the anomaly score. Such methods has an underlying assumption that, the latent space of the normal class can be better reconstructed in contrast to the anomalous instances. Advantages of using GANS's are, they are really good at generating realistic instances, especially on image data, enabling the network to see all the possible combinations of normal class representation. The disadvantages of GAN's like failure to converge and mode collapse are still manageable, but the major problem is the complex training schedule which makes the entire method very sensitive to the training domain. This also results in the sub-optimal anomaly score as they are based on the generator network which are designed for artificial data synthesis instead of anomaly detection.

2.2.3 *End-to-end Anomaly Score Learning*

In this approach, the learning methods aims to predict a scalar (anomaly score) in an end-to-end fashion. Such methods usually deduces a novel loss function, which drives the anomaly scoring network: $\tau(.;\Theta) : \mathcal{X} \mapsto \mathbb{R}$. More formally:

$$\Theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{x \in \mathcal{X}} \ell(\tau(X;\Theta)), \tag{2.4}$$

$$S_x = \tau(x, \Theta^*), \tag{2.5}$$

While such methods simultaneously learn the feature representation and the anomaly score, it is not limited by the inherited limitations of the above mentioned approaches. The common approaches are, Ranking models [79, 80], where a self trained model is used to regress the anomaly scores for unsupervised anomaly detection. The end-to-end anomaly scoring network takes a pseudo-anomaly and normal instance as the input and learns to optimize the anomaly score in way that the data input of similar class results in lesser score. Models also include to learn the latent dimension of the normal class and try to regress it to achieve the minimum score [1]. Some of the methods also tried to learn the one-class classifier on top of the deep network, making the entire setup as hybrid architecture[34]. The advantages of such methods are the anomaly scores, which can be directly optimised with novel adapted loss functions. The ranking models performs really good in limited anomalies scenarios, while may suffer to generalize the anomalies. The models can be adversarially optimized and trained in end-to-end fashion, there application is highly task dependent and sensitive to the training datasets.

All the methods described above are using either reconstruction based approach, or latent space, or GAN's, but they all are classifying the images on global level, either they are anomalous or normal. And most of the industries in the real life wants not only

to classify the images with the novelties but also they want to localize the anomalies in the images or at least an approximate area where the anomaly is located. While the obvious answer to this question can be a segmentation network; but the real industries have huge datasets,which are either poorly annotated or the industries doesn't want to spend resources on the annotation task. Apart from this high-imbalance data is quite ubiquitous in real life industrial scenarios. In addition to this they also want a fast network which can be easily be integrated into their present IT infrastructure.

Keeping in mind the industrial needs and the prospects offered by deep learning techniques, this dissertation explores multiple novel model for supervised, semi-supervised learning, for global image anomaly classification and localisation. The networks are capable of classifying images at global level, and also provide an approximate location for the anomalies in the images by scoring each pixel in the images.

3

Supervised Capsnet for Global Image Anomaly Classification

"Convolutional Neural Networks are doomed."
"The pooling operation used in convolutional neural networks is a big mistake and the fact that it works so well is a disaster."

– *Geoffrey Hinton*

Above are the two very famous quotes from *Geoffrey Hinton*, Turing Award [1] winner and often called as the "godfather of deep learning", from his lectures on computer vision as inverse graphics. So why does Hinton think that CNNs are bad?

In this chapter we will assess the pros and cons of Convolutional Networks. We will see how Capsule networks are filling the shortcomings of vanilla Conv networks. More

[1] https://en.wikipedia.org/wiki/Turing_Award

specifically we will see how an adapted capsule network, trained in a supervised fashion, with novel loss function can be used for global image anomaly classification.

3.1 What's so Common in Industrial Datasets?

Modern IT is playing humongous role in realising goals of the ambitious industries. Artificial intelligence is proving to be a game changer for production and manufacturing, enabling intelligent factory automation that is faster and cheaper than ever before [104].

These industries are producing Giga Bytes and Tera Bytes of data either through Internet-of-Things (IoT) devices or Visual Inspection Systems (VIS). However, most of the deep learning methods (either supervised, semi-supervised or unsupervised) suffer from the lack of generalization problem especially in case of imbalanced datasets. There are many industrial applications, where huge amount of normal data is present and it is possible to obtain labels easily, but the case of high class imbalance is still dominant in those industrial applications. Hence, training a neural network with highly imbalanced data is still challenging to the recent state-of-the-art deep neural networks. Keeping this problem in center, the proposed method addresses the anomaly detection task as the "*fully supervised*" classification problem under highly imbalanced datasets. For solving this problem we adopted *Capsule Networks* proposed by Hinton[99]. The network has shown some promising results by achieving state-of-the-art results on MNIST [60] dataset (0.25% test error). The capsule networks are the breakthrough as it provides a great improvement over the shortcomings of the traditional(vanilla) Convolutional Neural Networks (CNN).

3.2 Systems Architecture

Before we talk about the detailed system architecture of our proposed model, we will see the shortcoming of the CNNs and motivating the quotes by Hinton reported at the beginning of this chapter.

3.2.1 *Shortcoming of CNNs*

CNNs, achieved benchmarking results in image classification tasks like AlexNet [57], VGGNet[109], ResNet[41]. They have revolutionized the deep learning field. They have shown flexibility and great performance in the wider range of real life computer vision problems. But they are actually pretty bad in detecting objects in different posses [99]. Hence, to overcome this shortcoming either lots of training data is needed or techniques like data augmentation are used.

CNNs work on three main concepts which are, shared weights and biases, local receptive fields and activation, and pooling. As CNNs have additive scalar nature of neurons, at any given layer in the network they are ambivalent to the spatial relationship between the features captured at previous layers, and consequently within their effective receptive fields. Hinton et al.[99] introduced the idea of Capsules which solves the shortcomings of the traditional CNNs. For example an image like Figure 3.1 is enough to fool a simple CNN to believe that it's a good sketch of a human face. A simple CNN can easily extract the dominant features like nose, eyes, mouth correctly, but they fail or wrongly activate the neuron for the face detection.

3.2.2 *Capsule network advantages*

Invariance vs. Equivariance: A standard neural network such as a CNN has scalar outputs. For example, if the network is trained to discriminate between two classes such as cars and bikes, it will output two scalars, one associated to the car class and the

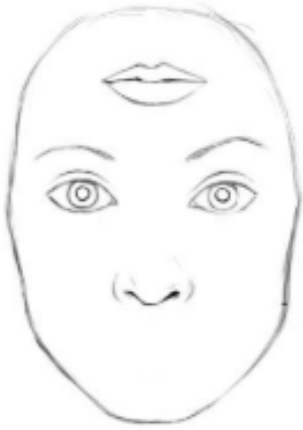


Figure 3.1: A wrong human face sketch to confuse a simple CNN.

other to the bike class: the highest one will determine the classification result. If the network is trained properly and can detect several variants of the objects belonging to the two classes, it means it reached some degree of *invariance* to input. For example, no matter if two cars have different visual aspects, they will both lead to the same output. On contrast, capsule network adopts a *vectorial outputs*. Each class is associated to an output vector, and the classification is done by looking for the vector with highest norm. In this case, two visually different cars could lead to very different output vectors, but they will still be classified as cars if the corresponding vector lengths are higher than the other outputs. This property is called *equivariance*, meaning that different input images from the same class will give different outputs, but they are equivalent. Equivariance is a desirable property because it allows to explicitly model different variants (Hinton calls them *poses*[99]) of the same object.

Equivariance is best understood by the way humans see and understand objects. Equivariance to a transformation means, if we take a input and perform some transformation on it, then the representation we get is the transformation of the representation of the originals.

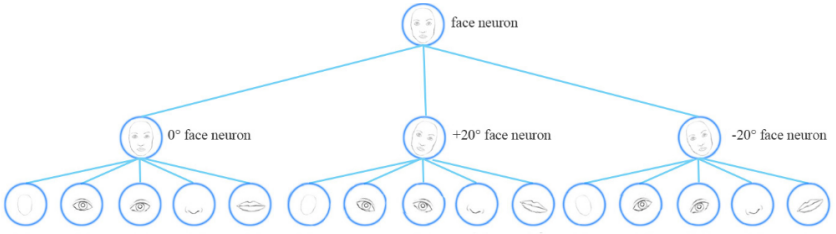


Figure 3.2: CNN neurons for detecting a face.

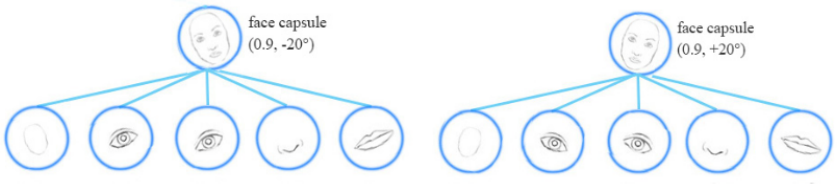


Figure 3.3: Capsule neurons for detecting a face.

Mathematically, Equivariance means -

$$transform(represent(y)) = represent(transform(y)) \tag{3.1}$$

while the invariance means -

$$represent(y) = represent(transform(y)) \tag{3.2}$$

Intuitively, a capsule network detects objects, that can be transform to each other, for example in the Figure 3.2 we can see that conceptually a CNN model uses multiple neurons and multiple stacked layers to match and then identify the different variants of the human face.

To summarize here are the major differences between the capsule and the neuron:

Feature coherence: Neural networks like CNN do not have an explicit way to model feature coherence. For example, let us consider the case of a face detection

Table 3.1: Major difference between a Capsule and a Vanilla Neuron

Capsule Vs. Traditional Neurons			
Input from low level capsules / previous layer neurons		**Vector (u_i)**	**Scalar (x_i)**
Operation	Affine Transformation	$\hat{u_j}\|i = W_i j u_i$	-
	Weighting		
	Sum	$s_j = \sum_i c_i j \hat{u_j}\|i$	$a_j = \sum_i w_i x_i + b$
	Nonlinear Activation	$v_j = \frac{\|\|s_j\|\|^2}{1+\|\|(s_j)\|\|^2} \frac{s_j}{\|\|s_j\|\|}$	$h_j = f(a_j)$
Output		vector(v_j)	scalar(h_j)

network. Intermediate layers of the network could specialize in detecting facial features like eyes, nose and mouth, and the presence of these three features could fire a face detection in the upper layers. However, this is independent of the relative spatial position of those features, and thus a simple CNN could be fooled by an image such as the one shown in figure 3.1. In contrast, capsule networks adopts vectorial outputs also for the intermediate layers (the capsules). An innovative routing-by-agreement algorithm then propagates their outputs to upper layers only if their vectors are coherent, meaning they have similar orientations. Combined with the equivariance property, this is a powerful classification scheme. For example, inner capsules could detect a nose and a mouth, and the training of the network will force their vectors to be coherent and activate a face detection in the upper layers. The same capsules could also detect a 30°-tilted mouth and nose (thanks to equivariance property) and the respective vectors could still be coherent to fire the detection of a tilted face. However,a tilted nose and a non-tilted mouth would lead to non-coherent vectors and the face would not be detected. This way, capsule networks explicitly enforce a global coherence of middle-level features in order to avoid false matches on images such as Figure 3.1, while Figure 3.3 shows that capsules can detect multiple variants of the face (i.e. if the face is rotated right 20° or rotated left 20°) rather than matching with the multiple variants of the face.

Because of these properties, we believe that capsule networks could lead to good results also when applied to anomaly detection problems. We thus adapted a standard CapsNet architecture to anomaly detection, as described in section 3.2.3.

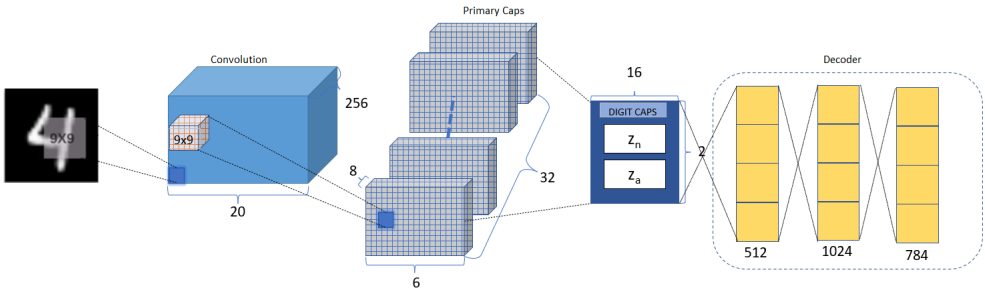


Figure 3.4: CapsNet architecture adopted for this study.

3.2.3 *Proposed Network*

The advantages mentioned in the previous sections is the major motivations to adopt the capsule network, especially the CapsNet architecture proposed by Hinton[99]. The original capsule network is developed for the MNIST digit recognition which consists of two main parts: an encoder that converts the input digit image into 10 vectors of instantiation parameters(digit caps), and a decoder which takes the longest vector out of the 10 vectors and reconstructs the original input. The network is optimized to maximize the vector length for the correct digit caps and to minimize the reconstruction loss. Although the decoder has minimal uses in the Hinton's[99] network, it is used to for the digit cap to adapt the meaningful instantiation parameters for the correct digit class.

In our proposed network [86] we are using all the basic architecture of the CapsNet, and we changed the number of digit caps to two, one for the normal class and the other for the anomaly class. Also, we are using the reconstruction of the image to perform image comparison with the original input and gain an extra hint of anomaly presence. The Figure 3.4 shows a schematic representation of our proposed network.

The main components of the network are:

- **Convolution:** It's a single vanilla convolution layer. This layer extracts the

features of the input image. It uses 256 kernels of the size 9x9 with activation as ReLU output.

- **PrimaryCaps:** This layer is also convolutional layer and it's output is 1152 feature vectors in $\mathbb{R}^8$. The squash operation, as defined in[99], preserves the orientation of the vectors while normalizes their length in the range (0,1).

- **Routing by Agreement:** Dominant features in vanilla CNN networks are extracted using the max pool operation. Routing by agreement is similar to that, but here it decides which information will travel to the next layer. In this process capsule tries to predict the next layer activation based on the vector length and the orientation of the detected object in the current layer. This algorithm is discussed in more detail in chapter 5

- **DigitCaps:** After routing-by-agreement, two digit caps are obtained. These are squashed vectors in $\mathbb{R}^{16}$, and capture the instantiation parameter for the each class (normal and anomaly class). The probability for each class is calculated by the length of the digitcaps vector, while its orientation represent the "pose". This specific orientation and the max length vectors are the final instantiation parameters among the many possible appearances for the same class.

- **Decoder:** The normal digitcaps vector is used for the reconstruction. The $\mathbb{R}^{16}$ vector is passed through the three fully connected layer to reconstruct the input image.

Our adapted network has 2 digits caps, while other backbone structure remains the same. However, in section 3.3, we will clearly reflect that the vanilla capsule network has extremely poor performance when the dataset is highly imbalanced. Our developed method uses two anomaly measures reconstruction loss and the vector length difference.

3.2.4 *The Novel Anomaly Scoring Method*

Here we explain how we use the reconstruction loss and the vector length difference to define our novel anomaly score, which will be subsequently used for the classification.

Reconstruction Loss: Reconstruction loss (r_l) is a L_2 loss, i.e. the Mean Squared Error, computed as the mean of the squared differences between the pixel values of the original and the reconstructed image. As the training data is highly imbalanced, the decoder network weights will be trained to reconstruct correctly only the normal class using the output of the normal digit caps. This ensures that the network will reconstruct correctly the normal class while it will behave poorly with the anomalous data. This technique is predominantly used in all the previous methods discussed in section 2, where anomalies are identified by their visual difference from their reconstruction created by a network trained only on normal data (semi-supervised learning).

Vector Length Difference: Since the proposed method is based on a full-supervised approach, we can also exploit the classification results, rather than just relying on reconstruction loss as in the semi-supervised approaches. In the proposed network architecture there are two digit capsules vectors (one for the normal class and the another for the anomaly class), we use the length of these vectors as the measure of the anomaly score. Let z_n and z_a be the two output vectors for normal and anomaly classes. The standard CapsNet architecture would classify an image as an anomaly if if $\|z_a\| > \|z_n\|$, but this approach does not give good results on imbalanced datasets (see the experimental results discussed in section 3.3). We thus investigated the values of the two outputs in presence of normal and anomalous data, and we discovered that the system behaves exactly as expected for the dominant class of normal images ($\|z_n\| \approx 1, \|z_a\| \approx 0$), while for the scarce class of anomalies the difference between the two vectors lengths is typically smaller. For example, vector lengths $\|z_n\| = 0.8$, and $\|z_a\| = 0.6$, are a strong indication that the sample is anomalous, even though the standard capsule net-

work would classify it in normal class. Hence, here we propose to use the vector length difference $\|z_a\| - \|z_n\|$ as a part of our final anomaly score.

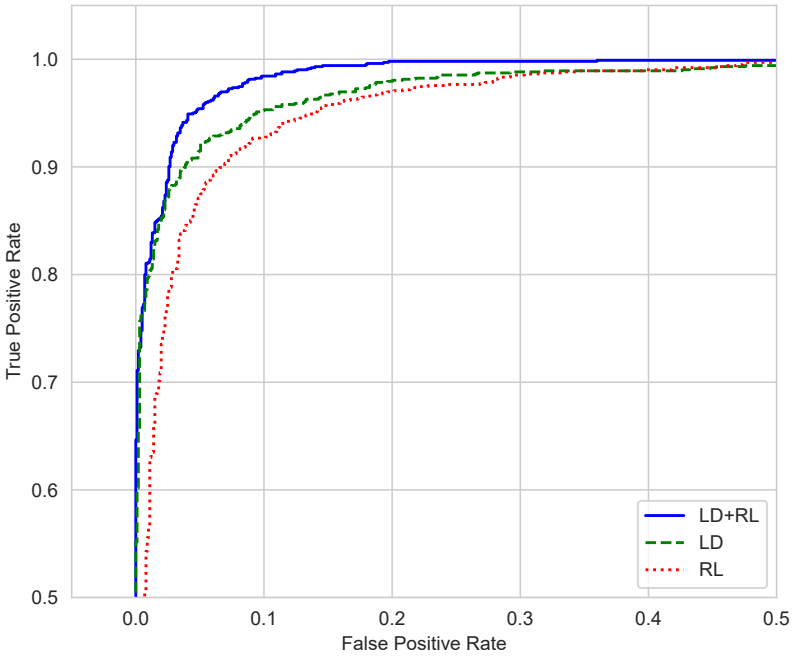


Figure 3.5: ROC curve for three anomaly detection measures: vector length difference, reconstruction loss, and vector length difference + reconstruction loss.

The final anomaly score AS is a combination of the two measures of reconstruction loss and vector length difference:

$$AS = \|z_a\| - \|z_n\| + r_l \tag{3.3}$$

with $\|z_a\|, \|z_n\|, r_l \in [0, 1]$. Figure 3.5 shows a ROC curve with the performance results using the two approaches alone and their combination. It is evident that the vector length difference approach performs better than the basic image comparison with re-

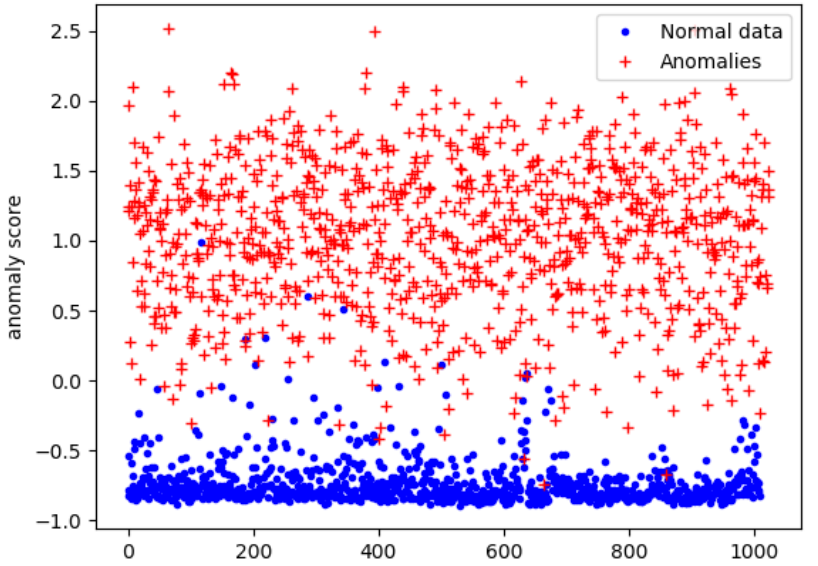


Figure 3.6: Anomaly scores on test data, training done with 10% anomalies. The test dataset is not imbalanced.

construction loss, thus justifying the adoption of a fully-supervised approach rather than a semi-supervised one. Moreover, the combination of the two anomaly scores leads to even better results, motivating the use of equation 3.3 as the proposed anomaly score.

Figure 3.6 shows the proposed anomaly scores on a MNIST test dataset for both normal (blue) and anomalous (red) data: it is visually evident that the two classes are well separated, giving a strong hint that the proposed metric is a good anomaly score. This will be measured rigorously in the next section.

3.3 Experimental results

The developed method has been tested with three datasets (MNIST[60], Fashion-MNIST[125], Kuzushiji-MNIST[26]). Each of the used datasets has 10 classes, respectively showing

digits, clothing, and ancient Japanese characters. The entire iterative scheme of training and testing follows these steps:

1. Choose a class as the normal class;
2. The training dataset contains all the training images of the chosen class plus some training images randomly picked from the other classes. The final dataset will be imbalanced, the total amount of anomalies varies depending on the experiment.
3. Train the network;
4. Test the system on a balanced test dataset.

This procedure is repeated for each class in the dataset, e.g. all the digits in the MNIST dataset. It should be noted that the test dataset is not imbalanced, this has been done to avoid biased results in the computation of the accuracy. The training hyperparameters are shown in table 3.2.

Table 3.2: Training hyperparameters

Adam learning Rate	0.001
% of anomalies in training data	1% – 10%
batch size	32
Epochs	10

3.3.1 *Datasets and Results*

- **MNIST Dataset**: MNIST dataset consists of 60,000 images of handwritten digits with size $28 \times 28 \times 1$. Since we use one class at a time as normal data, the training set is always composed of 6,000 normal images plus a variable amount of anomalies (10% or 1% of the normal data) randomly chosen among the other digits. The images have been standardized using the mean and variance of the whole dataset. Table 3.3 shows the achieved results with a standard CapsNet and our developed

Figure 3.7: Top rows: normal (left) and anomalous (right) samples from the MNIST test set. Bottom rows: the reconstructed images.

approach. Results of two cases 10% and 1% of anomalies in the training set are given. The result clearly shows that the standard CapsNet approach fails when the dataset is extremely unbalanced.

In particular, in the 1% case average accuracy of a standard CapsNet is 51.44%, which is close to random guess. On the other hand the proposed approach gives a high accuracy even with the extremely unbalanced datasets (accuracy is on average 98.84% and 96.46% for the 10% and 1% anomaly cases respectively). The reconstructed images for both the normal and the anomaly can be seen in Figure 3.7.

Table 3.3: Accuracy % on MNIST dataset for standard CapsNet and the proposed method. The amount of anomalies in the training data is 10% (top rows) or 1% (bottom rows).

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**avg**
Standard, 10% an.	97.46	98.78	97.02	92.87	96.36	93.42	96.87	96.83	95.50	92.13	95.72
Proposed, 10% an.	**99.50**	**99.27**	**99.22**	**99.21**	**99.10**	**98.33**	**98.74**	**98.05**	**99.00**	**97.93**	**98.84**
Standard, 1% an.	48.90	73.58	50.00	49.66	48.95	46.56	48.34	50.00	48.75	49.63	51.44
Proposed, 1% an.	**99.20**	**98.24**	**98.19**	**95.48**	**94.37**	**95.46**	**98.34**	**97.07**	**97.85**	**90.41**	**96.46**

- **Fashion MNIST Dataset**: Fashion MNIST dataset is similar to the MNIST dataset, except it contains the images from an online clothing store. It again contains 60,000 images for training and 10,000 images for testing, representing 10 classes of cloths. The images are 28×28 grayscale images and are standardized

similar to MNIST case. Labels encoding and Results are shown in table 3.5 and table 3.4 and reconstructions are shown in Figure 3.8. The datset is more complex than the digit MNIST dataset, but the results clearly shows that the proposed method outperforms the standard CapsNet when the training anomalies dataset is highly unbalanced.

Table 3.4: Fashion MNIST label encoding

Label	0	1	2	3	4	5	6	7	8	9
Desc.	T-shirt/top	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot

Table 3.5: Accuracy % on Fashion MNIST dataset.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**avg**
Standard, 10% an.	88.14	96.89	85.08	92.19	85.77	96.15	76.93	95.11	94.96	97.18	90.84
Proposed, 10% an.	**93.28**	**98.07**	**87.50**	**95.01**	**91.50**	**98.07**	**84.44**	**96.64**	**97.73**	**97.83**	**94.01**
Standard, 1% an.	49.41	49.41	49.41	49.41	49.41	49.41	49.41	49.46	49.41	49.41	49.41
Proposed, 1% an.	**87.45**	**95.31**	**84.98**	**90.86**	**87.70**	**94.27**	**77.32**	**93.33**	**92.14**	**96.15**	**89.95**

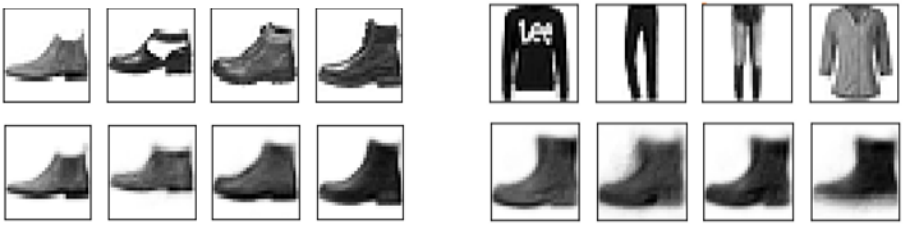


Figure 3.8: Top row: normal (left) and anomalous (right) samples from the Fashion MNIST test set. Bottom row: the reconstructed images.

- **Kuzushiji-MNIST(K-MNIST)**: This dataset contains the 28×28 grayscale images of ancient Japanese handwritten characters. The dataset size is same as MNIST and Fashion MNIST i.e. it contains the 60,000 training image and the 10,000 test images organized in 10 classes. It is a challenging dataset since images from the same class can be visually very different, as shown in Figure 3.9 where the 10 rows corresponding to each class can be seen. The accuracy for K-MNIST dataset can be seen in table 3.6 and reconstruction examples are in Figure 3.10.

The results obtained using this dataset confirm the robustness of our proposed approach. Our method outperforms the standard CapsNet architecture, especially in the extremely imbalanced dataset (1% training anomaly) case.

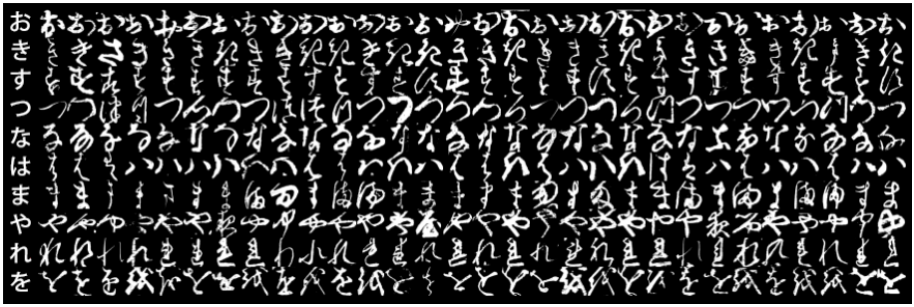


Figure 3.9: 10 classes of Kuzushiji-MNIST, with the first column showing each character's modern hiragana counterpart.

Table 3.6: Accuracy % on K-MNIST dataset.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**avg**
Standard, 10% an.	91.55	80.58	72.88	87.40	49.41	87.10	83.05	93.23	81.92	81.97	80.91
Proposed, 10% an.	**96.54**	**93.92**	**88.69**	**96.25**	**88.19**	**93.43**	**93.08**	**93.48**	**95.85**	**95.01**	**93.44**
Standard, 1% an.	49.95	49.95	49.95	49.95	49.95	49.95	49.95	49.95	49.95	49.95	49.95
Proposed, 1% an.	**92.31**	**86.11**	**79.17**	**93.06**	**83.27**	**90.81**	**86.56**	**76.37**	**87.91**	**90.71**	**86.63**

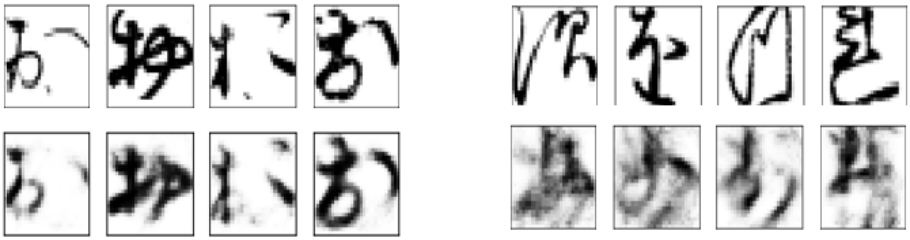


Figure 3.10: Top row: normal (left) and anomalous (right) samples from the K-MNIST test set. Bottom row: the reconstructed images.

The above experiments showed that the proposed method consistently outperforms standard capsnet approach in the case of 10% anomalies, and it performs well even in the 1% case, where the standard algorithm fails. We thus did further tests to compare the

two approaches and analyze their behaviour when the number of anomalies in the training dataset is particularly low. Figure 3.11 shows the achieved results on the MNIST dataset: as it can be seen, the proposed approach performs well even with extremely low amounts of anomalies (0.1% of the training dataset), while the standard capsnet error quickly grows when the anomalies are less than 6% of the training data.

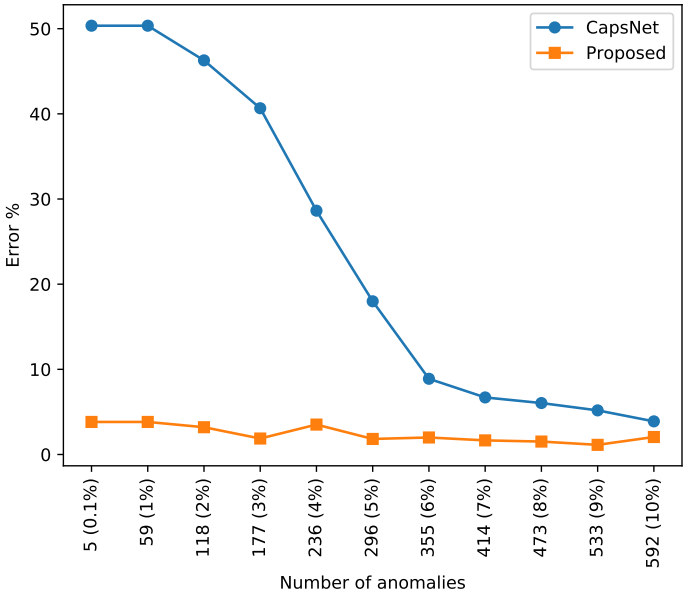


Figure 3.11: Comparison of standard CapsNet and the proposed method with varying amounts of outliers in a training set of 6000 samples (MNIST dataset, results are averaged over all the digits).

Finally, we compared the proposed system with other anomaly detection techniques. We first considered classical techniques such as Kernel Density Estimation and one-class Support Vector Machines both on PCA and Alexnet-extracted features, as well as Isolation Forests and Gaussian Mixture Models. Regarding more recent deep learning techniques, we considered both standard and variational autoencoders and two GAN-based models: AnoGAN[102] and ADGAN[29]. Results are taken from[29], where full details on the training parameters are given. Table 3.7 shows the achieved results, where

Table 3.7: Comparative results with other anomaly detection methods.

Dataset	y_c	KDE		OC-SVM		IF	GMM	DCAE	AnoGAN	VAE	ADGAN	Proposed
		PCA	Alexnet	PCA	Alexnet							
MNIST	0	0.982	0.634	0.993	0.962	0.957	0.970	0.988	0.990	0.884	**0.999**	0.995
	1	0.999	0.922	**1.000**	0.999	**1.000**	0.999	0.993	0.998	0.998	0.992	0.993
	2	0.888	0.654	0.881	0.925	0.822	0.931	0.917	0.888	0.762	0.968	**0.992**
	3	0.898	0.639	0.931	0.950	0.924	0.951	0.885	0.913	0.789	0.953	**0.992**
	4	0.943	0.676	0.962	0.982	0.922	0.968	0.862	0.944	0.858	0.960	**0.991**
	5	0.930	0.651	0.881	0.923	0.859	0.917	0.858	0.912	0.803	0.955	**0.983**
	6	0.972	0.636	0.982	0.975	0.903	**0.994**	0.954	0.925	0.913	0.980	0.987
	7	0.933	0.628	0.951	0.968	0.938	0.938	0.940	0.964	0.897	0.950	**0.980**
	8	0.924	0.617	0.958	0.926	0.814	0.889	0.823	0.883	0.751	0.959	**0.990**
	9	0.940	0.644	0.970	0.969	0.913	0.962	0.965	0.958	0.848	0.965	**0.979**
Average		0.941	0.670	0.951	0.958	0.905	0.952	0.919	0.937	0.850	0.968	**0.988**
CIFAR10	0	0.705	0.559	0.653	0.594	0.630	**0.709**	0.656	0.610	0.582	0.661	0.608
	1	0.493	0.487	0.400	0.540	0.379	0.443	0.435	0.565	**0.608**	0.435	0.555
	2	**0.734**	0.582	0.617	0.588	0.630	0.697	0.381	0.648	0.485	0.636	0.586
	3	0.522	0.531	0.522	0.575	0.408	0.445	0.545	0.528	**0.667**	0.488	0.587
	4	0.691	0.651	0.715	0.753	0.764	0.761	0.288	0.670	0.344	**0.794**	0.660
	5	0.439	0.551	0.517	0.558	0.514	0.505	**0.643**	0.592	0.493	0.640	0.501
	6	**0.771**	0.613	0.727	0.692	0.666	0.766	0.509	0.625	0.391	0.685	0.647
	7	0.458	0.593	0.522	0.547	0.480	0.496	**0.690**	0.576	0.516	0.559	0.484
	8	0.595	0.600	0.719	0.630	0.651	0.646	0.698	0.723	0.522	**0.798**	0.619
	9	0.490	0.529	0.475	0.530	0.459	0.384	**0.705**	0.582	0.633	0.643	0.576
Average		0.590	0.570	0.587	0.601	0.558	0.585	0.583	0.612	0.524	**0.634**	0.582

it can be seen that the proposed method on average outperforms the other techniques. For a fair comparison, it must however be noted that the proposed method is fully supervised although with imbalanced datasets (data shown in the table are from the 10% anomaly case), while the other techniques are either unsupervised or semi-supervised.

The problem of identifying anomalous images is still relatively new in the field of deep learning. Most approaches rely on autoencoders or GAN models to learn the aspect of normal images in a semi-supervised way, and identify anomalies by visual comparison. In this chapter we proposed an alternative approach based on fully supervised learning with imbalanced datasets. This idea came from real-world scenarios, in which anomalous data are often available but their amount is extremely scarce. The proposed approach, which is a variant of the the capsnet architecture, showed good performances even with extremely imbalanced datasets, outperforming both the standard capsnet architecture and other anomaly detection techniques.

As the proposed method is fully supervised approach, this brings us to the limiting condition of this approach. As discussed in section 3.1, the problems of modern indus-

tries, it's the need of hour that we develop semi-supervised or unsupervised approaches to deal with the anomaly detection problem. The next chapter we proposed a novel semi-supervised approach using stacked autoencoders and pyramidal feature extraction for global image anomaly classification.

4

Stacked Auto Encoders Using Pyramidal Features for Global Image Anomaly Classification

"The significance which is in unity is an eternal wonder."

– *Rabindranath Tagore*

Supervised anomaly detection methods are still significant and result quite well in real-life scenarios. But the limiting condition of the highly imbalanced dataset forces us to develop semi-supervised and unsupervised methods. In this chapter, we will discuss a semi-supervised approach using stacked autoencoders. The work learns the normality and in turn predicts the anomalous novel instances for higher losses.

4.1 Autoencoders: A Glance

Encoding-Decoding scheme or Autoencoders (AE) networks aim to learn low-dimensional feature representation space, which can further be well reconstructed. It consists of an encoding network and a decoding network. The encoder is used to map the high dimensional data (images in our case) to a low dimensional feature space (also called *latent space* [38]), while the decoder takes these latent space features and projects them to recover the original data. The bottleneck (latent space) is the most optimum low-dimensional representation of the original data. To minimize the overall loss of the recovered data via decoder, it is required that the retained features should be as much relevant as possible to the dominant instances, in our case normal instances. This retained information at the latent space is called *Causal Features*[38] This is a widely used technique of data compression or dimension reduction [42, 46]. The assumption in the anomaly detection scenario is that the network is unable to correctly reconstruct anomalous images, which can be identified by direct comparison of the original and reconstructed images. The basic formulation of this approach is given as follows:

$$z = \phi_e(\mathbf{x}; \Phi_e), \hat{\mathbf{x}} = \phi_d(\mathbf{z}; \Phi_d), \tag{4.1}$$

$$\Phi_e^*, \Phi_d^* = \arg\min_{\Phi_e, \Phi_d} \sum_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x} - \phi_d(\phi_e(\mathbf{x}; \Phi_e); \Phi_d)\|^2 \tag{4.2}$$

$$s_x = \|\mathbf{x} - \phi_d(\phi_e(\mathbf{x}; \Phi_e^*); \Phi_d^*)\|^2 \tag{4.3}$$

where ϕ_e is the network encoding with learnable parameters Φ_e and ϕ_d is the decoding network with the learnable parameters Φ_d. s_x is the anomaly score based on the reconstruction error of $\mathbf{x}$.

However, current methods generally do not address the problem at different scales.

Moreover, the comparison is often based on the trivial pixel-by-pixel comparison, which is not necessarily the best approach to evaluate image similarity[29, 2, 130, 97, 82]. Finally, many papers are evaluated on trivial datasets only, e.g. MNIST[60], which are not explicitly studied for anomaly detection problems.

4.2 Proposed Network

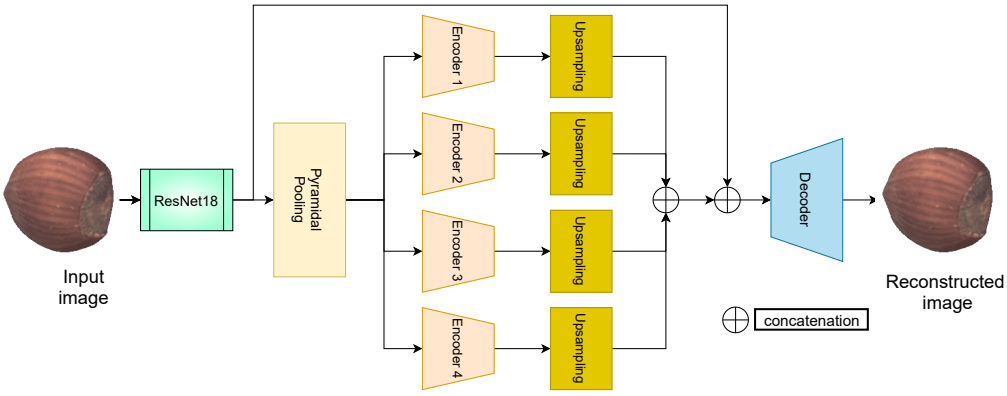


Figure 4.1: Proposed network architecture. The network consists of the first levels of a Resnet18 network for feature extraction, followed by a pyramidal pooling layer that feeds 4 encoders, each one connected to an upsampling layer with shared weights, and a final decoder. The features obtained from the four upsampling layers are concatenated with the output features of resnet18.

Our proposed model [69] runs around a significantly important question about representation learning: How to make one representation capture the causal factors of an image? One hypothesis to this problem is that features in any representation are the underlying causes of the ground truth data. Separate features or separately acquired features correspond to different causes and it's needed so that the representation can disentangle the causes from one another.

A representation of the ground truth that cleanly disentangles the causal factors are not always easy and most of the time our approach is to learn a procedure that

can easily learn a representation. However, the hypothesis also motivates the semi-supervised learning where capturing the causal factors in an objective. Hence, we are proposing a semi-supervised reconstruction-based approach to learn the causal features for anomaly detection. As we want to capture features that are more independent and are easier to capture, we are using a deep pyramidal representation of the causal features and using those features for the reconstruction-based approach.

Following a global trend in deep anomaly detection, we propose a reconstruction-based approach. The basic idea is to find a low-dimension feature representation of the input image that captures its fundamental properties (the *causal factors*, as named in some works) from which the image can be reconstructed. The network thus has an encoder-decoder structure, as in standard autoencoders, which ideally models an identity function, but passing through a dimensionality-reduction bottleneck after the encoding part. The main idea is that the network, when trained on normal data, learns a mapping from input space to the low-dimensional latent space which is suitable only for normal data. If anomalous data are fed as an input to the network, their reconstruction should be poor in quality, and thus the anomalies can be detected by image comparison with the original input.

Compared to other [1, 102, 29, 2] similar works, our main contribution consists in the addition of a pyramidal level in the network structure, in order to extract features at different resolution scales. This way we increase the chances to extract features at a scale level in which the anomaly is particularly evident. Another improvement consists in the way the input and reconstructed images are compared. Most methods rely on a trivial MSE loss that, when applied to image comparison context, consists of a simple pixel-by-pixel comparison. We instead propose a high-level perceptual loss, that better models visual similarity between images.

4.2.1 *Network architecture*

We propose a network to learn the manifold of the normal class by analyzing the feature representation at different scale levels using pyramidal pooling. This way the network can better find meaningful features that describe the image content at different scales, and as a consequence will perform better at detecting anomalies of different sizes.

Figure 4.1 shows the schematic diagram of our proposed novel network. The main components of the network include:

- **Resnet18** - A pre-trained Resnet18 network (trained over imagenet dataset) is being used for deep feature extraction from images. Only the first four layers of the network have been used. The basic idea is that the network can extract generic low-level features that are meaningful for many different types of images. excluding the maxpool layer, in the beginning, the average pool layer from the second last layer and the fully connected output layer. The output feature from the resnet block is in the shape (batch, 512, height, width).

- **Pyramidal Pooling Layer** - The pyramidal pooling layer scales the input features at different magnification levels, thus increasing the possibility that features relevant for the anomaly detection task is actually extracted, see Figure 4.2. The layer takes input from the Resnet 18 block and applies an average pooling such that the output will have a unit width and height. Then it uses a convolutional layer to reduce the channel features at different scales respectively 1, 2, 3, and 6, followed by batch normalization and ReLU activation layers. The outputs are respectively fed into four encoders. The bias from the convolution layer is made false because it's not desired to learn the average value, which allows to shift of the activation function while extracting the features of passed magnification.

- **Encoders** - We use four encoders, which receive their input from the pyramidal pooling layer. Each encoder is composed of a sequence of three convolutional

layers which reduce the input to a feature vector in $\mathbb{R}^8$. We used convolutional autoencoders without any regularization because the unsupervised training guarantees for the regularization [38]. Also, we didn't intend to use the variational autoencoders (VAE) because it seeks to converge representations to a fixed and expected value, whereas our solutions want to capture the different causal representations. This flexibility allows attuning the model reconstructing capability and the richness of latent representation. conversely, in VAE's, the effectiveness of fixed priors regularises and potentially leads to the over-smooth representation, which is not desirable for the anomaly detection tasks.

- **Up-sampling Layer** - The upsampling blocks are composed of two linear layers and are used to upsample the latent features of the encoders from $\mathbb{R}^8$ to $\mathbb{R}^{512}$. All the upsampling blocks share the same network weights. dimension 8 to dimension 512*mf(multiplying factor). The multiplying factor is calculated as the $0.5*$ (output height of Resent18 feature). We use four upsampling blocks for each encoder and shared weights. A detailed structure can be seen in table 4.1.

- **Decoder** - The decoder layer takes as input the concatenated features from upsampling layers and the output of the Resnet layer. The decoder uses 4 transposed convolutional layers to reconstruct the sample, thus giving in output an image of the same size as the input image. The detailed structure of the decoder can be seen in the table4.2.

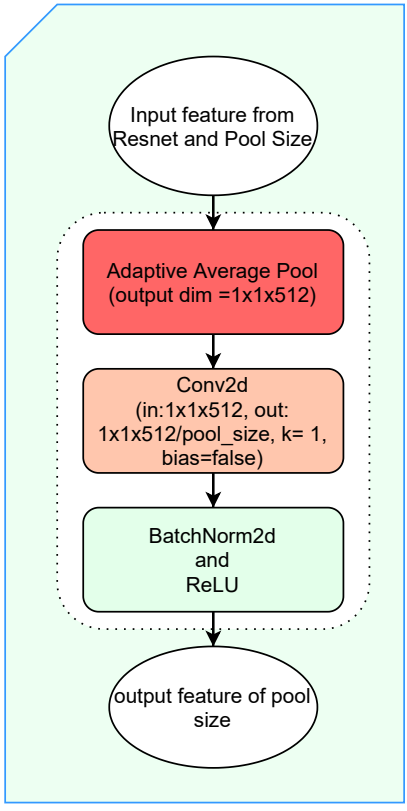
Each network layer is followed by a batch normalization layer and uses the ReLU activation function except for the last layer, where a sigmoid activation forces the pixel values to be in the range $[0, 1]$.

Table 4.1: Encoders and Up-sampling layer architecture: in,out,k,s,p means in_channel, out_channel, kernel, stride and padding respectively. 'mf' is multiplying factor which is calculated as 0.5*(output height of Resnet18 features)

Encoder1	**Encoder 2**	**Encoder 3**	**Encoder 4**	**Upsampling**
Conv2d in:512,out:16 ,k:3,s:1,p:1	**Conv2d in:256,out:16, k:3,s:1,p:1**	**Conv2d in:170,out:16, k:3,s:1,p:1**	**Conv2d in:85,out:16, k:3,s:1,p:1**	**Linear in:8,out:128**
Batch norm ReLU	*Batch norm ReLU*	*Batch norm ReLU*	*Batch norm ReLU*	*Batch norm ReLU*
Conv2d in:16,out:8, k:3,s:1,p:1	**Conv2d in:16,out:8, k:3,s:1,p:1**	**Conv2d in:16,out:8, k:3,s:1,p:1**	**Conv2d in:16,out:8, k:3,s:1,p:1**	**Linear in:128, out:512*** mf^2
Batch norm ReLU	*Batch norm ReLU*	*Batch norm ReLU*	*Batch norm ReLU*	*ReLU*
Conv2d in:8,out:8, k:1,s:1	**Conv2d in:8,out:8, k:1,s:1**	**Conv2d in:8,out:8, k:1,s:1**	**Conv2d in:8,out:8, k:1,s:1**	

Decoder	
MNIST and FM-NIST	**Mvtech**
ConvTranspose2d in:64,out:16:k:5,s:1p:1	ConvTranspose2d in:1024,out:16:k:3,s:2p:1
Batch norm ReLU	*Batch norm ReLU*
ConvTranspose2d in:16,out:32:k:5,s:1	ConvTranspose2d in:16,out:32:k:3,s:2p:1
Batch norm ReLU	*Batch norm ReLU*
ConvTranspose2d in:32,out:32:k:6,s:1	ConvTranspose2d in:32,out:32:k:4,s:2
Batch norm ReLU	*Batch norm ReLU*
ConvTranspose2d in:32,out:32:k:6,s:1	ConvTranspose2d in:32,out:3:k:4,s:2,p:1
Batch norm ReLU	*Tanh*
ConvTranspose2d in:32,out:3:k:5,s:1 1-1 *Tanh*	

Table 4.2: Decoder structure



The p

Figure 4.2: Proposed network architecture of pyramidal pooling block. The network consists of an adaptive average pooling layer withan output dimension of 1x1x512, a conv2d layer, batchnorm and relu. Input to this block is from the output feature of Resnet18 and pools size (four pool size 1, 2, 3 and 6 (magnification) has been used as input for encoder1, encoder2, encoder3, and encoder4 respectively.

4.2.2 *Objective and losses*

In order to train the network, we adopted a reconstruction-based approach, in which the network output is requested to be similar to the input. If the training is successful, it means that the low-dimension latent feature space in which the input is mapped after the encoders efficiently describes the visual properties of normal images. We assume that the same features will not be able to reconstruct anomalous images, which will then be identified by their higher loss. The network is trained using the following two

losses:

- *Reconstruction loss*: It's an MSE loss computed between the input and the reconstructed image, i.e. $\frac{1}{WH}\|X - \hat{X}\|_2^2$, where X is the input and $\hat{X}$ is the output of the network (reconstructed image). This is a pixel-by-pixel image comparison, widely adopted in other anomaly detection works. However, because of its intrinsic pixel-level independence assumption, it fails at modeling high-level visual features.

- *Perceptual loss*: Perceptual loss[49] is a more sophisticated loss trying to catch visually meaningful differences in images. It catches the high-level perceptual and semantic difference between images. It negates the assumption that all the pixels in an image are independent of each other. It is an MSE loss computed between the high-level image features obtained by a pre-trained VGG11 [109] network using its first four layers. The loss is defined as $\frac{1}{WHC}\|F(X) - F(\hat{X})\|_2^2$, where F is the transformation function applied through the trained four layers of VGG11 network, and W, C, H are the size of the resulting feature map. The trained network is only used for the calculation of loss and the weights are not updated during training.VGG11 network is used specifically as it has a very simple network structure without any maxpool layer, which makes it an ideal candidate for this work.

the proposed objective function minimizes the total loss L:

$$L = \frac{1}{WH}\|X - \hat{X}\|_2^2 + \lambda\frac{1}{WHC}\|F(X) - F(\hat{X})\|_2^2 \tag{4.4}$$

where X is the input image, $\hat{X}$ is the network output, and F is the non-linear function computed by the first four layers of a pre-trained VGG11 network. λ is a weighing factor between two losses, all the experiments discussed in section 4.3 are obtained with $\lambda = 1$.

4.3 Experimental results

We tested the proposed model on the publicly available standard datasets like MNIST [60] and FMNIST [125]. Although this dataset was not initially meant to be used in anomaly detection tasks, it has been widely adopted in literature to show the ability of the system to discriminate between one class, considered normal, and the other ones considered anomalies. In addition to this, we also tested our model over the recently published real-world anomaly detection dataset by MVTec [11], which contains more realistic data.

4.3.1 *Datasets and Results*

- **MNIST**: *MNIST* dataset consists of 60,000 28x28 gray images of handwritten digits, grouped in 10 classes. The gray images were converted to RGB images and then passed through the network. For training, one class has been considered as the normal class while the remaining classes are considered anomalies. Results are averaged over several runs in which each one of the original classes has been chosen as the normality model.

- **Fashion MNIST**: *FMNIST* dataset composed of 60,000, 28x28 grey images of clothes from an online clothing store. The images were standardized similar to the MNIST before passing to the network. Table 4.4 shows our network performs better in comparison to other state-of-the-art methods like GPND [87] and OCGAN [82]. The respective ROC curve score was used for the performance measurement.

- **MVTec** : MVTec dataset contains 5354 high-resolution color images of different texture and object categories (see Fig. 4.3). It contains normal and anomalous images (*70 different types of anomalies*) from real-world products. Since gray-scales are quite common in industrial uses, it has 3 object categories (*zipper,*

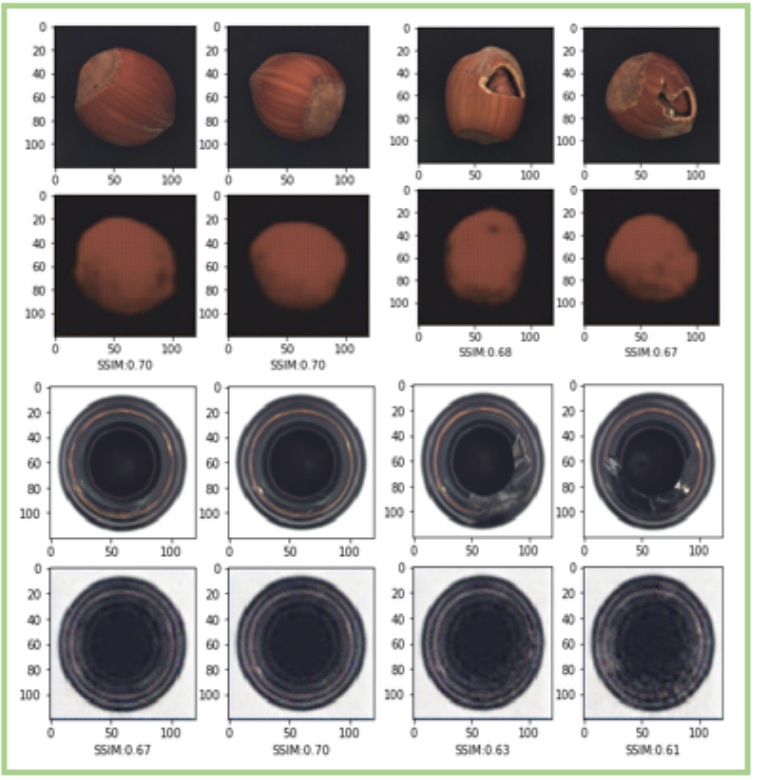


Figure 4.3: Examples from the MVTec dataset. The first two columns show the original and reconstructed images for normal objects (hazelnuts and glass bottles). The last two columns show the same results for anomalous images (broken hazelnuts, defective bottles).

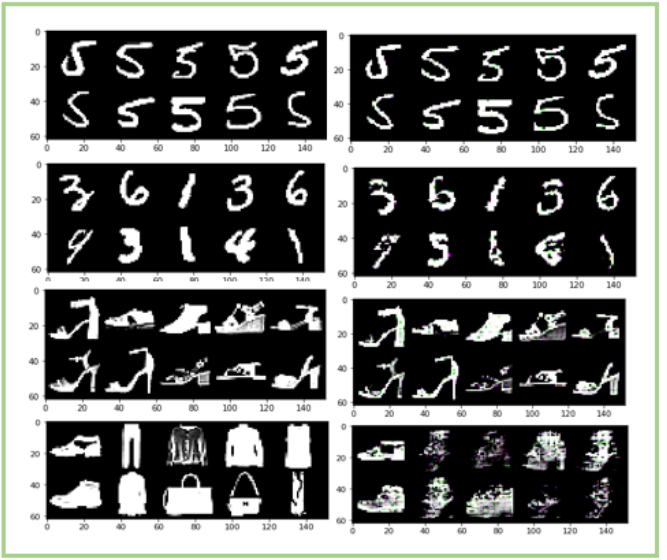


Figure 4.4: First Column: Ground truth for MNIST and FMNIST, Second Column: Reconstruction for MNIST and FMNIST. While the first and third row is for normal class, the second and last row is for anomaly class

screw, and grid) available solely in single-channel images. As the original image sizes were large, the images were resized to *120x120* pixels before passing it to the proposed network. The size justifies as it maintains the structural integrity of the images such that anomalies are still visible by the human eye.

We trained our model in order to learn the manifold of the normal class and force our decoder to capture all the causal features for the same. Training is started by initializing the weights of the network using *Orthogonal Initialization* [100], except the resent block, which was pretrained on imagenet, and the VGG11 block, which was pretrained on imagenet and was kept fixed.

The architectural hyper-parameters details are shown in the table4.3.

Table 4.5 shows the achieved results on the MNIST dataset. Tests have been done considering one of the classes as normal and using the remaining ones as anomalies, this

Adam learning Rate	0.0001
weight decay	0.0001
batch size	120
Epochs	600

Table 4.3: Training hyperparameters.

has been done for each one of the 10 classes. The achieved results have been compared with standard methods such as one-class support vector machines and kernel density estimators, as well as with deep learning approaches such as denoising autoencoders, variational autoencoders [52], Pix-CNN [115] and Latent Space Autoregression [1]. The comparative results have been taken from [1]. Performance is measured using the Area Under ROC Curve (AUC) metric. As it can be seen, the proposed method achieves the best result on 6 out of 10 classes, and it has the best average result, at the same level of LSA. Table 4.4 shows the achieved results on the FMNIST dataset. The test is done similarly to the MNIST approach. The achieved results are compared with the other state-of-the-art methods like GPND [87] and OCGAN [82]. Performance is measured using the AU ROC curve, averaged over 10 classes. The proposed methods perform at par and even better than the compared methods.

Table 4.6 shows the results on the MVTec dataset over all the 16 categories, comprising both textures (carpet, grid, leather, etc.) and objects (bottle, cable, capsule, etc). Our results are compared with other deep learning anomaly detection algorithms such as autoencoders with L2 norm loss and structural similarity loss [13], the GAN-based approach AnoGAN [102], and CNN feature dictionary [72]. The comparative results have been taken from [11]. Performance is measured again using True Positive Rate (TPR) and True Negative Rate (TNR). The study follows the same method as that of the compared state of the art. The proposed method achieves the best results on 9 out of 16 categories, and it reaches the best average result.

Table 4.4: ROC AUC for anomaly detection using FMNIST. We report the average value for the network for all the classes. Results are taken from [87, 82]

FMNIST	
Network	Average AU ROC
GPND	0.933
OCGAN	0.924
ours	**0.936**

Table 4.5: AUC results of anomaly detection using MNIST. Each row shows the normal class on which the model has been trained. Comparative results are taken from [1]

MNIST								
Class	**OC SVM**	**KDE**	**DAE**	**VAE**	**Pix CNN GAN**	**LSA**	**Deep SVDD**	**Ours**
0	0.988	0.885	0.991	0.994	0.531	0.993	0.98	**0.995**
1	**0.999**	0.996	**0.999**	**0.999**	0.995	**0.999**	0.0.997	**0.999**
2	0.902	0.710	0.891	**0.962**	0.476	0.959	0.917	0.941
3	0.950	0.693	0.935	0.947	0.517	**0.966**	0.919	**0.966**
4	0.955	0.844	0.921	**0.965**	0.739	0.956	0.949	0.960
5	0.968	0.776	0.937	0.963	0.542	0.964	0.885	**0.972**
6	0.978	0.861	0.981	**0.995**	0.592	0.994	0.983	0.992
7	0.965	0.884	0.964	0.974	0.789	0.980	0.946	**0.993**
8	0.853	0.669	0.841	0.905	0.340	**0.953**	0.0.939	0.895
9	0.955	0.825	0.960	0.978	0.662	0.981	0.0.965	**0.989**
Mean	0.95	0.81	0.94	0.97	0.62	**0.97**	0.948	**0.97**

Table 4.6: Results on the MVTec dataset. Each row shows the results achieved in a specific category. Each cell shows the best TNR (bottom) and TPR (top) values. The method with the highest mean of the two values is shown in bold. Comparative results are taken from literature [11].

MVTec Results					
Class	**AE SSIM**	**AE (L2)**	**Ano Gan**	**CNN Feat. Dict.**	**ours**
Carpet	**0.43** **0.90**	0.57 0.42	0.82 0.16	0.89 0.36	0.42 0.72
Grid	0.38 1.00	**0.57** **0.98**	0.90 0.12	0.57 0.33	0.86 0.53
Leather	0.00 0.92	0.06 0.82	0.91 0.12	**0.63** **0.71**	0.62 0.625
Tile	1.00 0.04	**1.00** **0.54**	0.97 0.05	0.97 0.44	0.44 0.85
Wood	0.84 0.82	1.00 0.47	0.89 0.47	0.79 0.88	**0.85** **0.95**
Bottle	0.85 0.90	0.70 0.89	0.95 0.43	1.00 0.06	**0.84** **1.00**
Cable	0.74 0.48	0.93 0.18	0.98 0.07	0.97 0.24	**0.58** **0.89**
Capsule	0.78 0.43	1.00 0.24	0.96 0.20	0.78 0.03	**0.62** **0.74**
Hazelnut	1.00 0.07	0.93 0.84	0.83 0.16	0.90 0.07	**0.90** **0.89**
Metal nut	1.00 0.08	0.68 0.77	0.86 0.13	0.55 0.74	**0.98** **0.55**
Pill	0.92 0.28	1.00 0.23	1.00 0.24	0.85 0.06	**0.76** **0.62**
Screw	0.95 0.06	0.98 0.39	0.41 0.28	0.73 0.13	**0.73** **0.71**
Toothbrush	0.75 0.73	**1.00** **0.97**	1.00 0.13	1.00 0.03	0.8 0.92
Transistor	1.00 0.03	0.97 0.45	0.98 0.35	1.00 0.15	**0.60** **0.89**
Zipper	**1.00** **0.60**	**0.97** **0.63**	0.78 0.40	0.78 0.29	0.64 0.82

4.3.2 *Ablation Study*

Here we propose a set of ablation studies, in which we removed some parts of the network and the whole setup is retrained in order to see the influence of those parts on the network performance. First, we try to study the effect of the number of encoders. We started it with one encoder and then successively in the multiple of 2 i.e. 1, 2, 4, and 6. We found that the learning performed better in terms of anomaly capturing and reconstruction capabilities. But these improvements got saturated as with the increase in the numbers of autoencoders. Ablation studies have been done on the MVTech dataset ('bottle' and 'carpet', one from product and one from texture category) so that complexity of the learning domain(accuracy) and the reconstruction capacity, Structural Similarity Index (SSIM) [43, 75, 24], can be tested. All the tests have been done with the hyperparameters, refer table 4.3, kept constant. The result for the same can be seen in Figure 4.5 and Figure 4.6.

While average SSIM obtained from configuration having 1 and 2 encoders remained below 0.65 for bottle and 0.40 for carpet, for the normal class, the average SSIM obtained with 4 encoders remained above 0.68 for bottle and 0.53 for carpet. Also, accuracy has been tested to choose the best model configuration. The accuracy with a configuration having 1 and 2 encoders remained below 85% and 36% for bottle and carpet respectively in comparison to 93+% and 57% obtained with 4 encoder configuration for bottle and carpet respectively. Distinctively the study showed that adding more AE didn't lead to statistically relevant improvement in the SSIM and accuracy of the model performance. As with the 6AE configuration, the results didn't improve much and remained pretty much the same. Hence, we choose 4 AE configurations for our further studies.

In addition to this, we also tried to apply the perceptual loss at the feature level i.e. between the concatenated out of the upsampling layer and the output of resnet block. This is done in order to map the features at both locations to represent the

same output of the resnet block. This will force the decoder for easy reconstruction. In the study, we found that neither MSE loss nor perceptual loss at the feature level was helpful. It minutely contributes to the early convergence of the training result and has no significant role while testing. Hence, we decided to drop the perceptual loss between the features.

At last, we also tried to study the effect of perceptual loss, see section 4.2.2, over the model performance. To measure the system performance two model has been trained in the following configuration: a) MSE loss only ($\lambda = 0$); b) MSE + Perceptual loss. The network was trained with 4AE configuration and constant hyperparameters, see table4.3, for 'Bottle' and 'Carpet'. The results as measured in terms of the AUC and can be referenced to Fig 4.7.

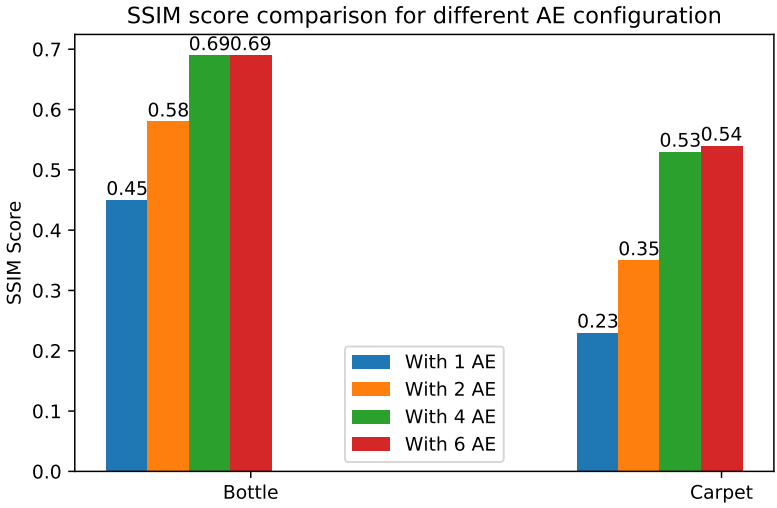


Figure 4.5: SSIM comparison results for different AE configurations.

our proposed deep network for the anomaly detection identifies anomalies by means of a network that encodes normal images in a low-dimensional latent space and then reconstructs them, ideally modeling an identity function. Since the network is trained

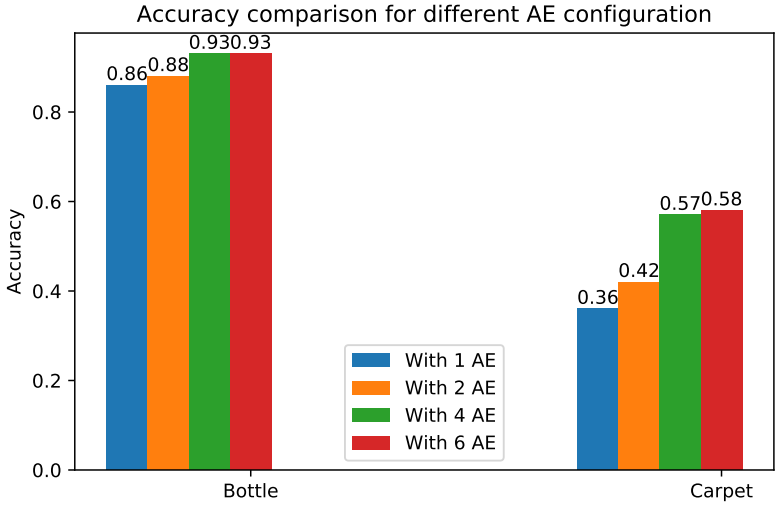


Figure 4.6: Accuracy comparison results for different AE configurations.

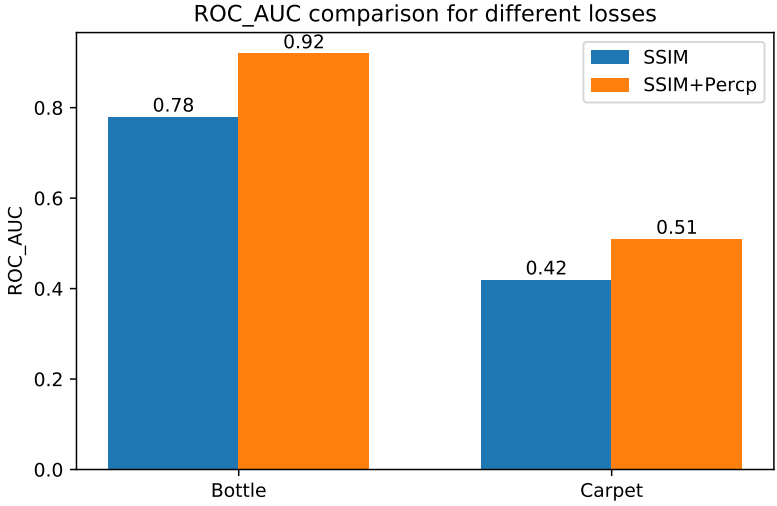


Figure 4.7: ROC AUC comparison for different losses.

on normal data only, it fails at reconstructing anomalous images, which can be detected by an image similarity loss. The main contributions of this work consist in the usage of a multi-scale pyramidal approach that extracts latent features at different resolutions,

and the usage of a high-level perceptual loss to better compare images at the feature level, rather than at pixel level. We also found that our proposed model worked best for the product while in the case of fine texture products like carpet, it further can be improved Moreover, differing from many works that have been evaluated on basic datasets only such as MNIST and FMNIST, we also tested the proposed network on MVTec, a real-world dataset of defective products. Achieved results are promising and often outperform other state-of-the-art methods.

While this work undertook a semi-supervised approach using stacked autoencoders. it still needs improvement as the network is still large, due to four AEs. Also, the network performed poorly for the texture images. In the next chapter, we again used a reconstruction based semi-supervised approach, which uses pyramidal features and *dynamic routing*, see chapter 3.2.3, for the global image level anomaly classification.

5

Pyramidal Image Anomaly Detector(PIADE)

"Life will not be a pyramid with the apex sustained by the bottom, but an oceanic circle whose centre will be the individual."
"From the heights of these pyramids, forty centuries look down on us."

– *Mahatma Gandhi*
- *Napoleon Bonaparte*

Semi-supervised or unsupervised approaches are lately the most used approaches for the anomaly detection tasks. Sometimes they are interchanged based on some intrinsic calculation which produces pseudo labels for the images. In this chapter, we talk about our new state-of-the-art method *Pyramidal Image Anomaly Detector* (PIADE) [68].

For this approach, we assume that there are no anomalies in the training set, thus

following a semi-supervised approach. The basic idea is to adopt a reconstruction-based strategy, in which a neural network learns how to encode the input images into a low-dimension latent space and then reconstructs them in order to minimize the difference between the original and reconstructed image. By training such a network on normal data only, the network will learn only the features that are useful for the reconstruction of the training set, and thus will fail at reconstructing anomalous images. The difference between input and reconstructed images can thus be used as an anomaly score.

5.1 Proposed Network

Despite the reconstruction-based approach has been already used in literature, in this chapter we propose PIADE (Pyramidal Image Anomaly DEtector), a network architecture that includes several novel strategies to improve the overall system performance. First, we adopt a pyramidal approach [111] that analyzes the image features at different scale levels in parallel. This way, we improve the probability that relevant image features are extracted at the scale level in which the anomaly is more evident. Then, we borrow the idea of *dynamic routing* from capsule networks [99] to have finer control on the features that are really useful for the task of anomaly detection. Moreover, most of the reconstruction-based methods use pixel-wise loss functions in order to compare reconstructions and input data. This assumes independence among the pixels, which is generally not true. Hence, we adopted a more sophisticated loss function that considers the inter-relationship between the pixels and a perception-based loss computed by a pre-trained network. Finally, while many papers [29, 2, 130, 97, 82] in this field have been evaluated on simple and not anomaly-oriented datasets such as MNIST [60], we tested our proposed network and method on one of the first real-world datasets for image anomaly detection published by MVTec [11]. We found that the proposed model performed at par or better when compared with various state-of-the-art methods.

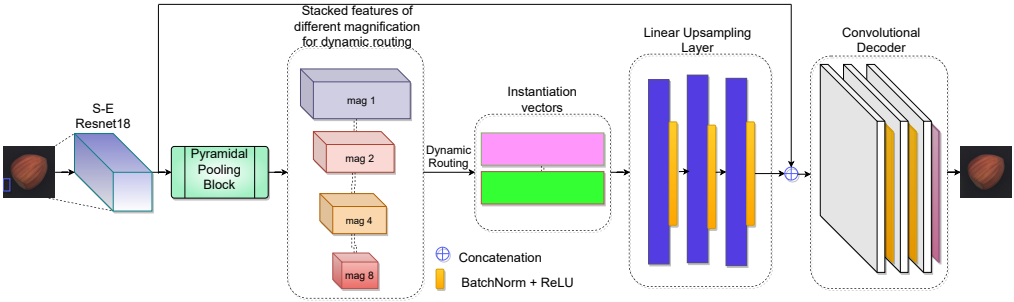


Figure 5.1: Proposed PIADE network architecture. The network consists of the first levels of a SE-Resnet18 network for feature extraction, followed by a pyramidal pooling layer that extracts features at different scales. The features are then dynamically routed to two instantiation vectors in $\mathbb{R}^{64}$. The vectors are passed to a linear upsampling layer and a final transposed convolutional decoder. The features obtained from the upsampling layer are concatenated with the output features of ResNet18 before passing to the final decoder.

5.1.1 *Network Architecture*

The PIADE network architecture is shown in Figure 5.1. It consists of an initial ResNet block to extract basic image features. These features are then pooled in the pyramidal pooling block, in order to represent them at different scales. Following the idea of capsule networks [99], the pooled features are then dynamically routed to two instantiation vectors (more details below), in order to filter the best ones that are useful for later image reconstruction. Image is then reconstructed via a linear upsampling layer and a convolutional decoder in order to obtain an output with the same shape as the input data. We here give a detailed description of each block.

- **SE-ResNet18** - A pre-trained ResNet18 network is used for deep feature extraction. The network was trained over the imageNet dataset [31]. All 5 convolutional layers of the ResNet18 [41] have been used. The primary idea is that the network is able to extract generic features of images. A pre-trained network also gives the benefits of transfer learning as the real-world datasets are mostly related to the imageNet dataset.

In order to improve the quality of the extracted features, each convolutional block is followed by a Squeeze-and-Excitation (SE) block, as proposed by Hu et al. [45] (see Figure 5.2). The SE block is a form of attention mechanism among convolutional channels, that adaptively calculates channel-wise weights to explicitly model the interdependencies between channels.

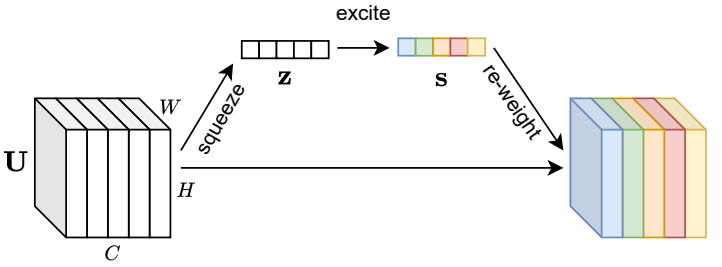


Figure 5.2: Squeeze-Excitation (SE) Block [45].

Formally, given the output tensor $\mathbf{U} \in \mathbb{R}^{W \times H \times C}$ of a convolutional layer, SE first squeezes it to a vector $\mathbf{z} \in \mathbb{R}^{1 \times 1 \times C}$ by aggregating the spatial dimensions according to the following equation:

$$z_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j). \tag{5.1}$$

Then, the excitation vector $\mathbf{s} \in \mathbb{R}^{1 \times 1 \times C}$, containing the channel weights, is computed as follows:

$$\mathbf{s} = \sigma(\mathbf{W}_2 \delta(\mathbf{W}_1 \mathbf{z})) \tag{5.2}$$

where σ is an element-wise sigmoid function, δ is a ReLU activation function, and $\mathbf{W}_1 \in \mathbb{R}^{\frac{C}{4} \times C}, \mathbf{W}_2 \in \mathbb{R}^{C \times \frac{C}{4}}$ are two learned matrices that model the excitation function itself. Then, the tensor $\mathbf{U}$ is channel-wise re-weighted using weights $\mathbf{s}$ (meaning that each channel $\mathbf{U}_c \in \mathbb{R}^{W \times H}$ is multiplied by the scalar $\mathbf{s}_c$) to generate the output of the SE block, which is subsequently passed to the other

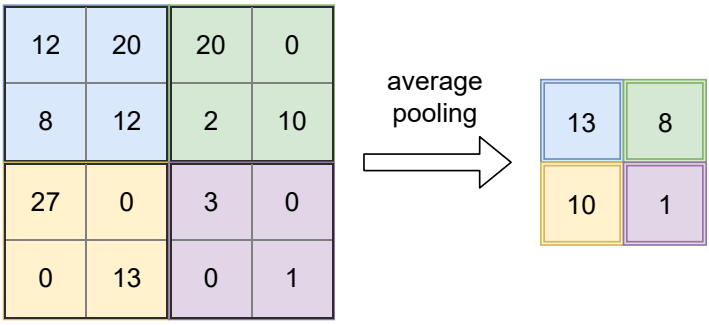


Figure 5.3: Average pooling with $k = 2$.

network layers. The entire operation can be seen as a self-attention mechanism on the channels using global information of the entire receptive field.

- **Pyramidal Pooling Layer** - The idea behind the pyramidal pooling layer is that image features can be analyzed at different magnifications, and possibly relevant features that are well-visible at a given scale could be not well extracted by the network at another scale. The pyramidal pooling layer thus scales the input features at different magnification levels, thus increasing the possibility that features relevant for the anomaly detection task is actually extracted.

  Each pyramid layer consists of the application of an adaptive average pooling. For a given feature map with spatial size $W \times H$, standard average pooling creates a new map with size $W/k \times H/k$, and each element of the new map is the average of the corresponding elements in the original map, as shown in Figure 5.3. Adaptive average pooling works in a similar way, but the term k is automatically chosen to guarantee a fixed size output, independently from the input size.

  In the proposed architecture we adopt four different pyramid levels, respectively with spatial output sizes of 8×8, 4×4, 2×2, and 1×1. The number of channels is left unchanged (in our case, the ResNet18 module final output uses 512 channels). Each pooling layer is followed by a convolution to reduce the number of channels

(1×1 kernel, stride=1, 64 channels) and batch normalization. The output features of each pyramid level are then flattened, concatenated, and reshaped in a final set of 8-dimensional feature vectors.

- **Dynamic Routing** - Dynamic routing is a novel algorithm proposed by Hinton in his Capsule Networks paper[99]. In capsule networks, the belonging of a sample to a given class is represented by a vector (called instantiation vector) rather than a scalar value. The norm of the vector represents how much the sample belongs to a specific class, while the vector itself represents a specific instance of that class, hence the name. Each instantiation vector is defined as a sum of several features, and dynamic routing is the algorithm that dynamically chooses which features must be routed to each vector. In other words, for each class, dynamic routing selects the best features that are more suited to describe that class.

  Dynamic routing algorithm is shown in algorithm 1, where the `squash`() function forces the vector norms to be in the range $[0, 1]$:

$$\texttt{squash}(s_j) = \frac{||s_j||^2}{a + ||s_j||^2} \frac{s_j}{||s_j||} \tag{5.3}$$

  In the proposed system, dynamic routing is used to route the features from the pyramidal pooling layer to two instantiation vectors, described below.

Algorithm 1 Dynamic Routing Algorithm [99]

function ROUTING($\hat{u}_{j|i}, r, l$)
▷ $\hat{u}_{j|i}$ are the features from layer i to j
▷ l is the current layer
    $\forall i \in l, \forall j \in l+1 : b_{ij} \leftarrow 0$
    for r iterations **do**
        $\forall i \in l : c_i \leftarrow \texttt{softmax}(b_i)$
        $\forall j \in (l+1) : s_j \leftarrow \sum_i c_{ij} \hat{u}_{j|i}$
        $\forall j \in (l+1) : v_j \leftarrow \texttt{squash}(s_j)$
        $\forall i \in l, \forall j \in (l+1) : b_{ij} \leftarrow b_{ij} + \hat{u}_{j|i} \cdot v_j$
    end for
    return v_j
end function

- **Instantiation Vectors** - In a standard capsule network classifier, each instantiation vector represents a class. In our case, we adapted this model to anomaly detection by using only two instantiation vectors: the first one represents the normal class, while the second one is just used to collect all the features that are discarded by the routing algorithm because not relevant enough to model the normal class. Observe that this is not the same as modeling the anomaly class because we do not have anomalies in the training set. In the proposed reconstruction-based method, these vectors are the low-dimension ($\mathbb{R}^{64}$ in our experiments) latent space in which images are mapped before reconstruction. For this reason, during training, the first instantiation vector is always passed to the upsampling and decoding layers. While testing, the vector with the maximum norm is decoded instead. This method is in contrast with traditional approaches where all the features contribute to the output computation, which can be accomplished using a single instantia-

tion vector (and thus no routing at all). In section 5.2.3 we propose an ablation study to prove that the two-vectors approach always performs better than the single-vector approach.

- **Upsampling and decoding layers** - The upsampling layer consists of three linear layers and it is used to upsample the instantiation parameters from $\mathbb{R}^{64}$ to $\mathbb{R}^{512\times mf\times mf}$, where "mf" is the multiplying factor equals to the width of output features from SE-Resnet18. The decoder is made of transposed convolutional layers [107, 62], which take concatenated features from the SE-Resnet18 and upsampling layers with dimensions $\mathbb{R}^{batch\times n\times 8\times 8}$ and transforms them into the reconstructed image of the same size as the input image.

5.1.2 *Objective and Losses*

As stated before, the proposed model uses a reconstruction-based approach, in which the aim is to produce a network output similar to its input. Since the model is trained over the normal class (single class, semi-supervised training), it is imperative that the causal features for the normal class are learned at the instantiation vectors. Subsequently, taking the maximum length vector for the reconstruction at the time of testing, it's assumed that the total reconstruction error at the time of testing will be higher. In order to measure the similarity between the original image and its reconstruction, we considered three possible loss functions:

- *MSE Loss* - Mean Squared Error (MSE) loss is a pixel-level loss, which assumes independence between pixels. MSE loss is computed as the average of the squared pixel-wise differences of the two images, and can be formally defined in terms of the Frobenius norm as $\frac{1}{WH}\|X-\hat{X}\|_F^2$, where X is the input and $\hat{X}$ is the output of the network (respectively the original and the reconstructed image), and W, H are the image width and height. MSE loss is often used in many reconstruction-

based works, however, the pixel-level independence assumption is unrealistic in real-world images.

- *Perceptual Loss* - Perceptual loss [49] is a more sophisticated loss trying to catch visually meaningful differences in images. It is an MSE loss computed between the high-level image features obtained by a pre-trained VGG11 network [109] using its first four layers. The loss is defined as $\frac{1}{WHC}\|F(X) - F(\hat{X})\|_F^2$, where F is the transformation function applied through the trained four layers of VGG11 network, and W, C, H are the size of the resulting feature map. The trained network is only used for the calculation of loss and the weights are not updated during training.

- *Structural Similarity Index* - The Structural Similarity Index (SSIM) [13] is used to measure the image similarity by considering visual properties that are lost in the standard MSE approach. The important feature in this loss calculation is that it takes care of perceptual phenomena, including both luminance and contrast, and it is defined as:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{5.4}$$

  where, μ_x, μ_y, are the average values of input and reconstruction image, σ_x^2, σ_y^2 are the variance of input and reconstructed image, σ_{xy} is their co-variance and c_1, c_2 are the two constants used for numerical stability.

The overall proposed objective function minimizes the total loss L, defined as a weighted sum of the three image comparison losses:

$$L(X, \hat{X}) = MSELoss(X, \hat{X}) + \lambda_1 PercLoss(X, \hat{X}) + \lambda_2 SSIM(X, \hat{X}) \tag{5.5}$$

where X is the input image and $\hat{X}$ is the image reconstructed by the network. λ_1 and λ_2 are weighing factors between three losses. All the experiments discussed in section 5.2

are obtained with $\lambda_1 = \lambda_2 = 1$; since we noted that small differences in these values do not lead to significant differences in the results. In the ablation study presented in section 5.2.3 we show the results with $\lambda_1 = 0$ and $\lambda_1 = \lambda_2 = 0$, thus disabling the perceptual and/or the SSIM components.

5.2 Experimental Results

In this section, we present the experimental results obtained with PIADE. We first describe the datasets used for training and testing, then we describe the testing procedure and the adopted metrics to measure the system performance. Finally, comparative results are given, in order to evaluate the achieved results with other state-of-the-art works on anomaly detection.

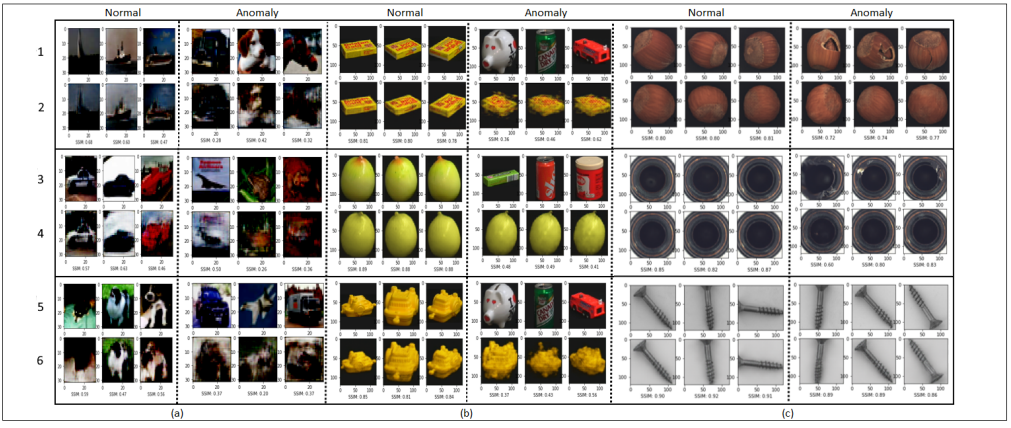


Figure 5.4: Reconstructions of normal and anomaly images for (a) CIFAR10, (b) COIL-100, and (c) MVTec datasets. Rows 1, 3, and 5 show few examples taken from the normal class and from the anomalies. Rows 2, 4, and 6 show the reconstructed images. The network is unable to correctly reconstruct anomalous data. The normal classes shown for the CIFAR10 examples are respectively Ship (row 1), Car (row 3), and Dog (row 5). The normal classes for COIL100 are the first three objects of the dataset. The normal classes for MVTec are Hazelnut, Bottle, and Screw.

5.2.1 *Performance metrics*

In order to measure the system performance, we consider the total loss $L(X, \hat{X})$ (equation 5.5) as a measure of the degree of the anomalous image. This is a sound approach since the loss measures the dissimilarity between input and reconstructed images, and the reconstruction-based approach assumes this dissimilarity will be high for anomalous images.

Once this anomaly score is computed, a threshold is required to convert it to a binary classification: any image with a score above the threshold will be considered anomalous, while the remaining ones will be classified as normal. Once this classification is done, standard statistics such as True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN) can be computed. These raw values are then converted into suitable ratios: in particular, we considered the True Positive Rate (TPR, also known as Recall, or Sensitivity) and the False Positive Rate (FPR), defined as follows:

$$\begin{aligned} TPR &= TP/(TP + FN) \\ FPR &= FP/(FP + TN) \end{aligned} \tag{5.6}$$

Rather than choosing an arbitrary threshold, we followed the popular approach of computing the (TPR, FPR) pairs for every possible threshold: the plot of these values gives the well-known Receiver Operating Characteristic (ROC) Curve. Finally, the area under the ROC curve (AUC) is used as a performance measure that summarizes the overall quality of the achieved results. This approach has been adopted to perform comparative analysis with state-of-the-art methods both on CIFAR10 [56] and COIL-100 [73] datasets. Observe that the choice of (TPR, FPR) pairs, and thus the use of ROC curves, is typically more suited when the tested dataset is balanced, while on imbalanced datasets the Precision/Recall values are more suited since they are not affected by the large abundance of True Negatives. However, despite dealing with anomalies, our

datasets are balanced for testing: in CIFAR10 and COIL-100, anomalies belong to 9 classes out of 10, and thus can be safely chosen so that their total amount is comparable to the amount of normal data.

Also, the MVTec dataset [13] has enough test anomalies to be considered balanced for testing. In this case, however, we adopted the same testing procedure described in the original paper where the dataset is proposed[11]. Here, for each class and each tested method, we compute the Sensitivity (TPR) and Specificity (TNR) for all the possible thresholds, and we select the best pair (TPR_{best}, TNR_{best}) as the one that maximizes their sum $TPR + TNR$. The best algorithm for each class is defined as the one with the highest mean of the two values, i.e. with the highest $(TPR_{best} + TNR_{best})/2$.

5.2.2 *Datasets and Results*

The proposed PIADE model has been tested using publicly available datasets. Tests have been done on CIFAR10[56] and COIL-100 [73] datasets. Although these datasets are not specifically designed for anomaly detection tasks, they are useful to show the ability of the system to discriminate between one class, considered normal, and the other ones considered anomalies. Since the performances of many previous works have been evaluated on these datasets, testing on CIFAR10 and COIL-100 is important for comparative results. In addition to this, the proposed model has also been tested on the real-world MVTec anomaly detection dataset [11], which is a recently published dataset specifically for anomaly detection tasks (see figure 5.4).

- **CIFAR10**: It contains 60,000 images with size 32×32 pixel. Images are grouped into ten classes: Airplane, Automobile, Bird, Cat, Deer, Dog, Frog, Horse, Ship, and Truck. 50,000 images are for training while 10,000 images are for testing. For this study, we treated one of the classes as normal and the rest as an anomaly. The results presented here are averaged over all the classes in several runs, in which

each one of the original classes is chosen to be the normality model.

- **COIL-100**: The dataset has been taken from the Columbia Object Image Library. It contains 7,200 color images of 100 objects, having dimension 128×128 pixels. Differing from CIFAR10, each class in COIL-100 represents a single object, but is observed from different points of view: each object was kept on an automated turntable, and the images were taken at a fixed pose step of 5 degrees. For each object, a total of 72 images were recorded. As in the case of CIFAR10, experimental results are averaged over 100 runs, each one with a different class chosen as normal. While training, one of the objects is treated as normal while all others as anomalies. The images were resized to 120×120, this is to maintain the same network structure used for the MVTec dataset. As the number of images in this dataset is limited, we used the training strategy of Pidhorskyi et al. [87], and split the training and testing data with a ratio of $80\% : 20\%$.

- **MVTec Dataset**: MVTec recently published a real-world anomaly detection dataset. It contains 5,354 high-resolution color images of different textures and objects categories. It has normal and anomalous images which showcase 70 different types of anomalies of different real-world products. It also contains 3 products images in grayscale, as grayscale images are very common in industrial practices. With this dataset, all the images were first resized to 120×120 pixel size before being passed to the model. Image anomalies are still visible at this resolution.

The model is trained by minimizing the total loss $L(X, \hat{X})$ (equation 5.5). The same loss is also used in the testing phase as an anomaly score. The model weights are initiated with orthogonal initialization except for the ResNet block, which was pre-trained on imageNet, and the VGG11 block, which was pre-trained on imageNet and kept fixed. The architectural hyper-parameters are shown in the table5.1. The model

has been implemented in Python using the Pytorch framework [81],[1], the source code is available online[2]. All the experiments have been done on a dual Xeon E5-2660 CPU, 224 GB RAM, 1 Tesla K40, and 2 Titan XP GPUs. On such hardware, training took on average 20 minutes for each class of the MVTec dataset, and 30 minutes for each class of the CIFAR10 dataset. Inference time is however extremely low, requiring on average 0.012 seconds to classify a single MVTec image and 0.006 seconds for a CIFAR10 image.

Table 5.1: Training hyperparameters.

Adam learning Rate	0.001
weight decay	0.0001
batch size	120
Epochs	400

Table 5.2: AUC scores using the COIL-100 dataset

Models	**Reference**	**AUC**
GPND	*NIPS 2018*	0.979
OCGAN	*CVPR2019*	0.995
DCAE	*MLSDA14*	0.908
PIADE (ours)	–	**0.998**

Table 5.3 shows the results obtained for the CIFAR10 dataset. Training has been done considering one class as normal and the rest as an anomaly, and this procedure has been repeated over all the classes. The achieved results have been compared with standard methods such as one-class support vector machines and kernel density estimators, as well as with deep learning approaches such as denoising autoencoders, variational autoencoders [52], Pix-CNN [115], and Latent Space Autoregression [1]. The comparative results have been taken from the work by Abati et al. [1]. Table 5.3 shows that the proposed model superseded the results of the state-of-the-art models in 7 out of 10 classes, and it has the best average result.

[1] `https://pytorch.org/`
[2] `https://github.com/pankajmishra000/PIADE`

Table 5.3: AUC scores using the CIFAR10 dataset. Each row shows the normal class on which the model has been trained. Comparative results are taken from literature [1]

Class	OC SVM	KDE	DAE	VAE	Pix CNN	GAN	LSA	PIADE (ours)
0	0.630	0.658	0.718	0.688	0.788	0.708	0.735	**0.751**
1	0.440	0.520	0.401	0.403	0.428	0.458	**0.580**	0.550
2	0.649	0.657	0.685	0.679	0.617	0.664	0.690	**0.708**
3	0.487	0.497	0.556	0.528	0.574	0.510	0.542	**0.609**
4	0.735	0.727	0.740	0.748	0.511	0.722	0.761	**0.805**
5	0.500	0.496	0.547	0.519	0.571	0.505	0.546	**0.645**
6	0.725	**0.758**	0.642	0.695	0.422	0.707	0.751	0.729
7	0.533	0.564	0.497	0.500	0.454	0.471	0.535	**0.651**
8	0.649	0.680	0.724	0.700	0.715	0.713	0.717	**0.771**
9	0.508	0.540	0.389	0.398	0.426	0.458	**0.548**	0.532
Mean	0.586	0.610	0.590	0.586	0.551	0.592	0.641	**0.675**

Table 4.6 shows the results on the MVTec dataset over all the 16 categories, comprising both textures (carpet, grid, leather, etc.) and objects (bottle, cable, capsule, etc). Our results are compared with other deep learning anomalies detection algorithms such as autoencoders with L2 norm loss and structural similarity loss [13], the GAN-based approach AnoGAN [102], and CNN feature dictionary citenapoletano2018anomaly. The comparative results have been taken from the work by Bergmann et al. [11]. Performance is compared by computing the average of the best TPR and TNR for each class and for each model. The proposed method achieves the best results in 10 out of 15 categories. It is worth noting that PIADE performs poorly on the texture classes (Carpet, Grid, Leather, Tile). This is probably due to the ResNet module, which has been pretrained to extract features that are meant to describe full objects rather than textures and patterns.

5.2.3 *Ablation Studies*

We here propose a set of ablation studies, in which the network is re-trained after the removal of specific parts in order to measure the influence of those parts on the network

Table 5.4: Results on the MVTec dataset. Each row shows the results achieved in a specific category. Each cell shows the best TNR (top) and TPR (bottom) values. The method with the highest mean of the two values is shown in bold. Comparative results are taken from literature [11]

Class	**AE SSIM**	**AE (L2)**	**Ano Gan**	**CNN Feat. Dict.**	**PIADE (ours)**
Carpet	**0.43** **0.90**	0.57 0.42	0.82 0.16	0.89 0.36	0.33 0.74
Grid	0.38 1.00	**0.57** **0.98**	0.90 0.12	0.57 0.33	0.67 0.60
Leather	0.00 0.92	0.06 0.82	0.91 0.12	**0.63** **0.71**	0.82 0.45
Tile	1.00 0.04	**1.00** **0.54**	0.97 0.05	0.97 0.44	0.97 0.19
Wood	0.84 0.82	1.00 0.47	0.89 0.47	0.79 0.88	**0.85** **0.93**
Bottle	0.85 0.90	0.70 0.89	0.95 0.43	1.00 0.06	**0.87** **1.00**
Cable	0.74 0.48	0.93 0.18	0.98 0.07	0.97 0.24	**0.73** **0.93**
Capsule	0.78 0.43	1.00 0.24	0.96 0.20	0.78 0.03	**0.60** **0.83**
Hazelnut	1.00 0.07	0.93 0.84	0.83 0.16	0.90 0.07	**0.89** **0.97**
Metal nut	1.00 0.08	0.68 0.77	0.86 0.13	0.55 0.74	**0.57** **0.78**
Pill	0.92 0.28	1.00 0.23	1.00 0.24	0.85 0.06	**0.89** **0.51**
Screw	0.95 0.06	0.98 0.39	0.41 0.28	0.73 0.13	**0.80** **0.67**
Toothbrush	0.75 0.73	**1.00** **0.97**	1.00 0.13	1.00 0.03	0.92 0.96
Transistor	1.00 0.03	0.97 0.45	0.98 0.35	1.00 0.15	**0.70** **0.90**
Zipper	1.00 0.60	0.97 0.63	0.78 0.40	0.78 0.29	**1.00** **0.65**

performance.

The first ablation study has been done to justify the use of dynamic routing with two instantiation vectors. Our hypothesis is that using a single vector, and thus disabling dynamic feature routing, will lead to worse results, since all the features are forced to

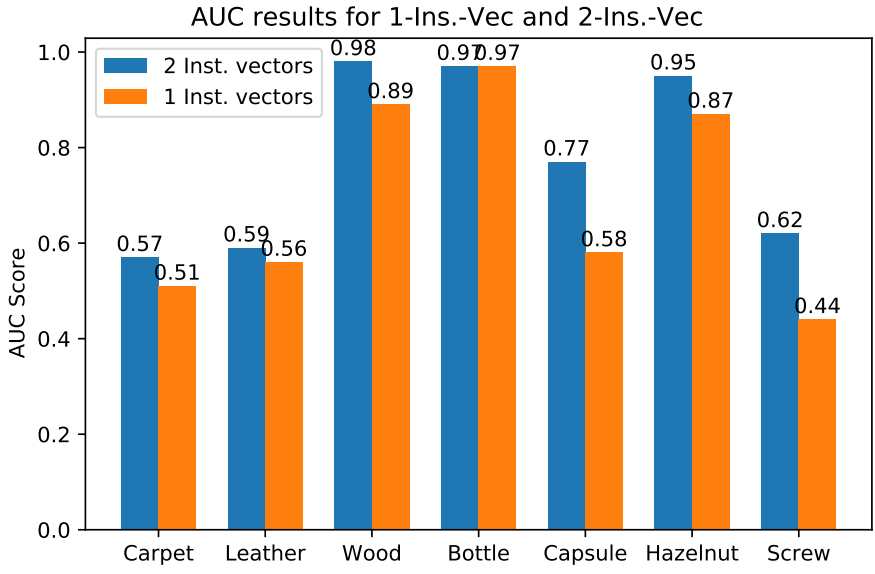


Figure 5.5: Comparison of AUC for one and two instantiation vectors respectively.

contribute to the same instantiation vector. On the other hand, with two vectors, features are allowed at testing time to accumulate at the second vector if they do not give a valid contribution to the image reconstruction task. The proposed model reconstruction capabilities have been tested by comparing the Structural Similarity Index (SSIM) of the reconstructed images. The ablation study has been made using the MVTec dataset, maintaining the same hyperparameters used for regular testing (Table 5.1). For the comparison, three products (*Bottle, Capsule, Hazelnut*) and three textures(*Carpet, Leather, Wood*) have been chosen from the dataset. The comparative results can be seen in Figure 5.5. In all the categories, the two instantiation vectors approach performed better than one vector with a 9% improvement on average.

Another ablation study has been done to study the effect of the Squeeze-Excitation attention module. In this case, we used the ResNet with and without soft attention and found that the model with soft attention performed better in comparison to vanilla

ResNet. We took the same 7 objects of the previous experiment and measured the AUC. Results of SE-ResNet performed always better or at par with the vanilla ResNet, as shown in Figure 5.6.

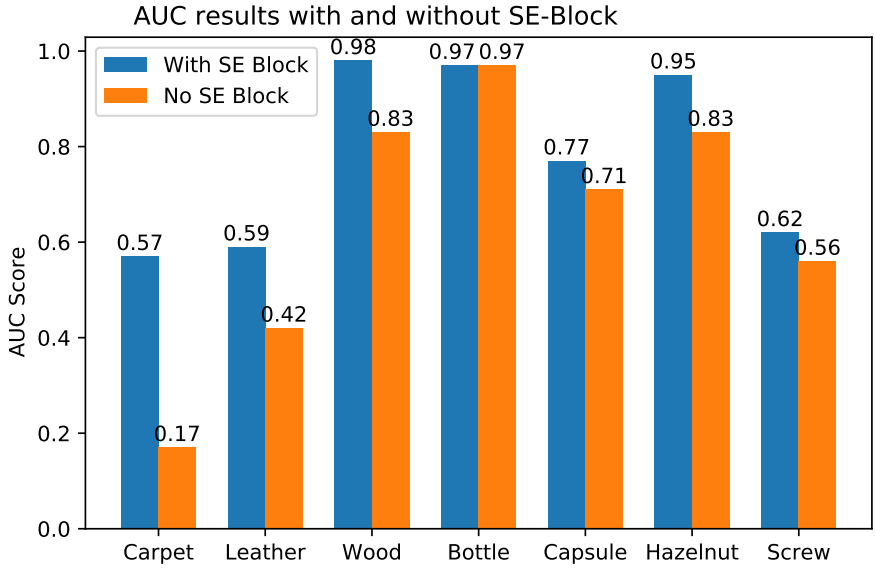


Figure 5.6: Comparison of AUC with and without the Squeeze-Excitation attention module.

The last ablation study aims at testing the performance of the three-loss functions described in section 5.1.2. In order to measure the system performances, three models have been trained with the following configurations: a) MSE loss only ($\lambda_1 = \lambda_2 = 0$); b) MSE + SSIM loss ($\lambda_1 = 0$); c) MSE + SSIM + Perceptual loss ($\lambda_1 = \lambda_2 = 1$). The network was trained on the MVTec dataset categories "bottle", "capsule" and "carpet", results are measured in terms of AUC. The results are shown in Figure 5.7 clearly show that the model with all the three losses has superior results if compared to the other configurations.

Hence, in this work, we proposed a reconstruction-based, semi-supervised deep neural

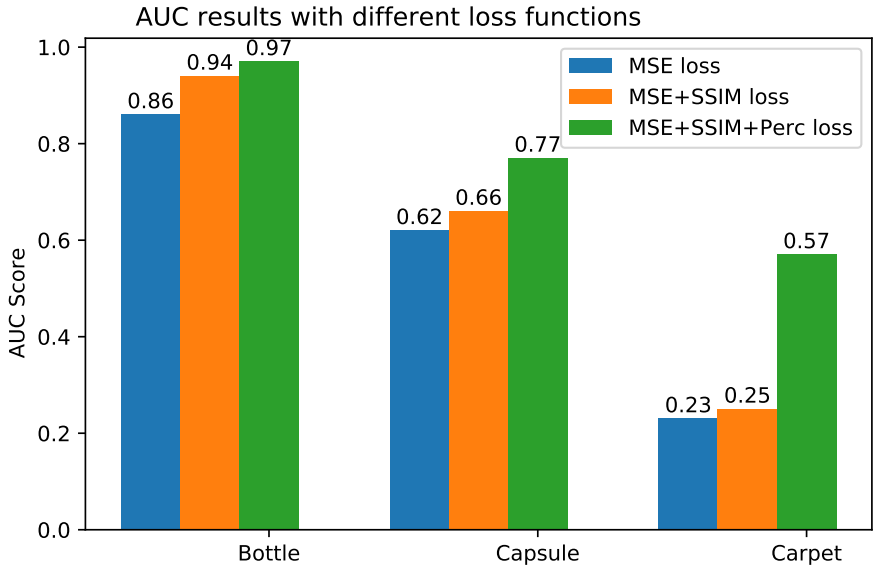


Figure 5.7: Comparison of AUC with different loss function combinations.

network for image anomaly detection. The network is trained on normal data only, and it builds a "normality model" by mapping the input images in a low-dimension feature space, from which they can be correctly reconstructed. The inability of the network to reconstruct other images allows the identification of anomalies, which can be detected by their higher reconstruction error. Compared to other state-of-the-art works, the proposed models include a pyramidal multi-scale approach to analyze image features at different scale levels, a dynamic routing layer inspired by the architecture of capsule networks, and a high-level image comparison loss. Moreover, the system has been tested not only on standard datasets such as CIFAR10 and COIL-100 (which have not been initially created for anomaly detection experiments) but also on the recently proposed MVTec dataset of anomalies in industrial images. Experimental results showed that the proposed model is at least at par, and often outperforms other state-of-the-art works. Further ablation experiments prove the validity of the architectural choices on which

the proposed network is based.

Till now we have seen the methods, which are capable of classifying in real-life industrial scenarios. While this gives rise to another interesting paradigm, where a model can also localize the anomaly in the images. In the next chapter, we will see our novel method, where a single model is capable of both classifying an image globally as normal or anomaly while localizing the anomaly in the image. The most interesting part is while doing this, it doesn't need any ground truth or pixel precise mapping of anomalies for the training.

6

VT-ADL : A Vision Transformer Network for Image Anomaly Detection and Localization

"Do Not Forget What You Have Learned Of Our Past, Rodimus. From Its Lessons, The Future Is Forged."

– *Optimus Prime (* ***Transformers: The Movie****)*

Learning the local features and keeping the positional information in an unsupervised way is a novel and very challenging task in the computer vision domain. The most informative part of any image dataset is the part with the highest variance. While such regions can be located in a very small as well as large local area in an industrial

image. Hence, extracting those pixel-precise region, in an unsupervised way and at the same time preserving the spatial information is often desirable, as it helps in automating various industrial scenarios. Most of the classical machine learning techniques try to learn this variance (also termed as *Entropy* [38]) in the dataset, and with the rise in modern IT infrastructure there is a boom in data acquisition, which subsequently requires better and efficient algorithms to be processed.

6.1 Transformer: A Glance

Since the paper "Attention is all you need" [116] in 2017, the way attention was thought has changed. The paper showed that with enough data, matrix multiplications, linear layers and normalization layers one can propose state-of-the-art language models. It is this technology which is behind the famous language models like BERT [32], GPT-2 [90], GPT-3 [15], which were the state-of-the-art solutions for various tasks, including language modeling, text summarizing, and question answering. In year 2020 transformers foray into the computer vision field [20]. And ViT [33] is the most successful application of transformers in computer vision tasks.

Before taking transformers to the anomaly detection task and computer vision applications, let's first see some basic structural ideas from the language models. We will be seeing some basic structure of the transformers, which will help us to understand the functioning better.

6.1.1 *Transformers in NLP: A Brief Overview*

The Figure 6.1is taken from the original paper of Vaswani et al. [116], it shows the encoder-decoder architecture of the model, being left and right respectively. Both encoder and decoder are composed of *Multi-Layer Attention* and *Feed Forward Layers* modules, which are stacked on top of each other. Here we try to look closer at these

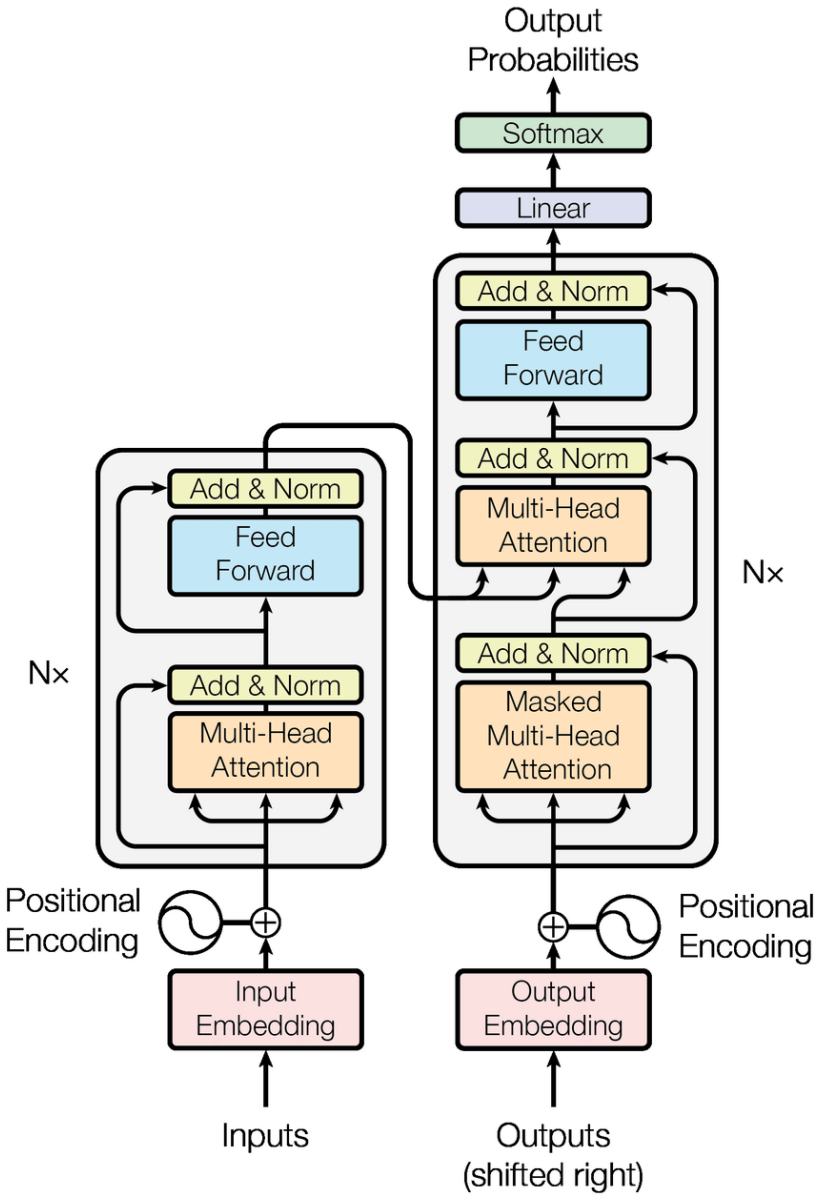


Figure 6.1: The Original Transformer-model architecture from Vaswani et al. [116]

Multi-Head-Attention modules (see Figure 6.2) of the model as these are the most important part in transformer functioning.

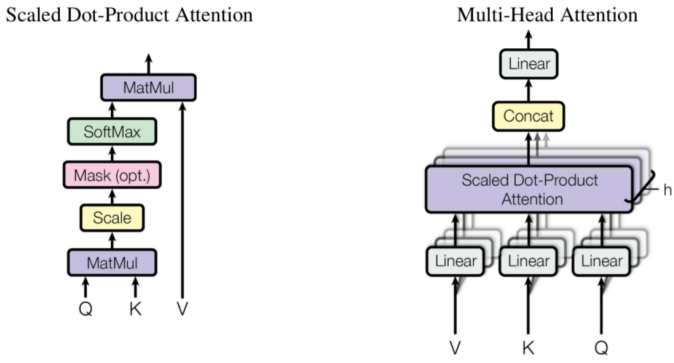


Figure 6.2: left shows Scaled Dot-Product Attention, right shows Multi-head Attention layers running in parallel, from Vaswani et al. [116]

Let's see the left part of the Figure6.2, it basically shows the equation 6.1 :

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{6.1}$$

Q is called *Query matrix*, which contains the vectorized representation of one word in the sequence. K is called *Keys matrix*, which contains the vector representation of all the words in the sequence. V is called *Value matrix*, which is again the vector representation of all the words in the sequence. For the encoder and decoder, Multi-head attention module, V is the same word sequence as Q. But, for the attention module that is taking into account the encoder and the decoder sequence, V is different from the sequence represented by Q. Mathematically, we can say that attention calculated is the weighted multiplication of V. Where the weights a is given by :

$$a = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \tag{6.2}$$

It means that the weighing factor a is calculated with the dot product of Q and K, meaning, how each word of the sequence (matrix Q) is influenced by all the other words

in the sequence (matrix K). The softmax function projects all the weights a to have a distribution between 0 and 1.

The right hand of the Figure 6.2 shows the attention mechanism in parallel. The parallelization is achieved with the linear projection of Q, K, and V. This allows the system to learn from the combinations of Q, K, and V, which further helps in model generalization.

Similarly, in the computer vision application, these word token sequence is obtained from the image patches. Motivated from the benefits of transformers and industrial needs, we developed a *Vision-Transformer-based Image Anomaly Detection and Localization network* (VT-ADL), which learns the manifold of normal class data in an unsupervised way, thus requiring only normal data in the training process.

6.2 Proposed Method

In our work, we propose a novel Deep Anomaly Detection (DAD) network using adapted transformer network and Gaussian approximation [14, 114] for anomaly classification and localization in industrial images and also how different configurations can be tweaked to win some of the shortcomings of the vision transformer network. We tried to discover the limitation of transformer networks in the vision field and tried to mitigate them. Additionally, use of only linear layers in the transformer network makes it really fast to train and test, and this makes it a competitive candidate for industrial uses.

In addition to this, we also published a real-world industrial dataset (the beanTech Anomaly Detection dataset — BTAD) for the anomaly detection task. The dataset contains a total of 2830 real-world images of 3 industrial products showcasing body and surface defects.

The proposed model combines the traditional reconstruction-based methods with the benefits of a patch-based approach. The input image is subdivided into patches and

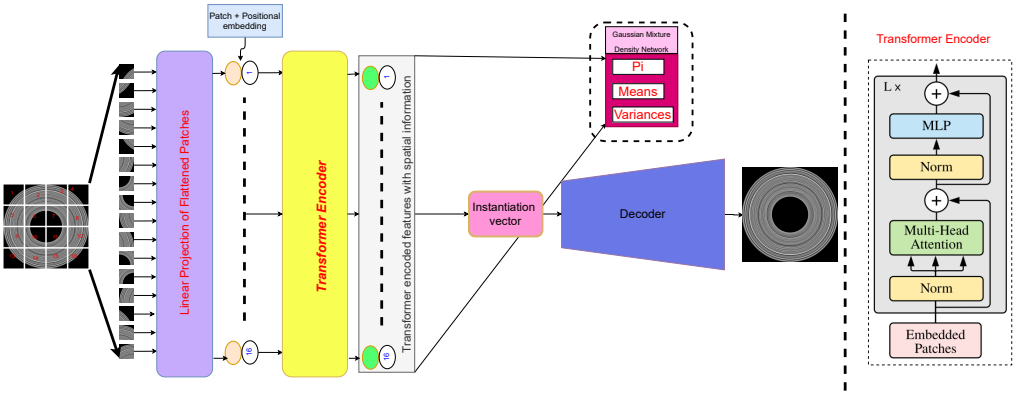


Figure 6.3: Left image: a model overview. Image is split into patches, which are augmented with positional embedding. The resulting sequence is fed to the Transformer encoder. Then encoded features are summed into a reconstruction vector which is fed to the decoder. The transformer encoded features are also fed into a Gaussian approximation network [14], which is later used to localize the anomaly. Right image: detailed structure of the transformer encoder (image from [33]).

encoded using a Vision Transformer. The resulting features are then fed into a decoder to reconstruct the original image, thus forcing the network to learn features that are representative of the aspect of normal images (the only data on which the network is trained). At the same time, a Gaussian mixture density network models the distribution of the transformer-encoded features in order to estimate the distribution of the normal data in this latent space. Detecting anomalies with this model automatically allow their localization, since transformer-encoded features are associated with positional information.

An overview of the model is depicted in Figure 6.3. To handle a 2D image $X \in \mathbb{R}^{H \times W \times C}$, we break the image into a sequence of 2D patches $X_p \in \mathbb{R}^{N \times (P \times P \times C)}$, where (H, W) is the original image resolution, C is the number of channels, (P, P) is the patch dimensions and N is the resulting number of patches $N = HW/P^2$. These patches are then embedded to a D-dimensional embedding space through a linear layer. Positional embedding is added to the patch embedding to preserve the positional information.

- **Transformer Encoder**: The transformer encoder layer is based on the work by Vaswani et al [116] and its application to images by Dosovitskiy et al [33]. The input patches are first mapped to the embedding space and are augmented with positional information (eq. 6.3), then passed to a Multi-headed Self-Attention block (eq 6.4) and an MLP block (eq. 6.5). Layer normalization (LN) is applied before the two blocks and residual connections are added after the two blocks. We didn't use the dropout layer throughout the network, as this causes instability in the Gaussian approximation network. MLP contains two linear layers with a GELU activation function.

$$Z_0 = [X_p^1\mathbf{E}; X_p^2\mathbf{E}; ...; X_p^N\mathbf{E}] + \mathbf{E}_{pos}, \tag{6.3}$$

where $\mathbf{E} \in \mathbb{R}^{(P^2.C)\times D}$, $\mathbf{E}_{pos} \in \mathbb{R}^{(N+1)\times D}$

$$Z_l^{'} = MSA(LN(Z_{l-1})) + Z_{l-1}, l = 1..L \tag{6.4}$$

$$Z_l = MLP(LN(Z_l^{'})) + Z_l^{'}, l = 1..L \tag{6.5}$$

The final encoded patches are reshaped and projected in to a reconstruction vector via learned projection matrix.

- **Reconstruction Vector**: It's a vector of dimension 512, which is obtained by a learned projection matrix. The projection matrix projects all the patch embedding from the normal instance, coming from encoder to a vector of dimension $\mathbb{R}^{512}$.

- **Decoder**: The decoder is used to decode the reconstruction vector back to the original image shape. It maps $\mathbb{R}^{512} \longrightarrow \mathbb{R}^{H\times W\times C}$. In our experiments with the MVTec and BTAD dataset, we used 5 transposed convolutional layers, with batch

normalization and ReLU in-between, except for the last layer, we use tanh as the final non-linearity.

- **Gaussian Mixture Density Network**: This kind of network estimates the conditional distribution $p(y|x)$ [14] of a mixture density model. In particular, the parameters of the unconditional mixture distribution $p(y)$ are estimated by the neural network, which takes the image embedding (conditional variable x) as the input. For our purpose, we employ the Gaussian Mixture Model (GMM) with full covariance matrix Σ_k as the density model. The density estimate $\hat{p}(y|x)$ follows the weighted sum of K Gaussian functions.

$$\hat{p}(y|x) = \sum_{k=1}^{K} w_k(x;\theta)\mathcal{N}(y|\mu_k(x;\theta), \sigma_k^2(x;\theta)) \tag{6.6}$$

wherein, $w_k(x;\theta)$ denotes the weight, $\mu_k(x;\theta)$ the mean, $\sigma_k^2(x;\theta)$ the variance of the k-th Gaussian. All the GMM parameters are estimated using the neural network with parameters θ and input x. The mixing weights of the Gaussians must satisfy the constraints $\sum_{k=1}^{K} w_k(x;\theta) = 1$ and $w_k(x;\theta) \geq 0\ \forall k$. This is achieved using the softmax function to the output of weight estimation:

$$w_k(x) = \frac{exp(a_k^w(x))}{\sum_{k=1}^{K} exp(a_i^w(x))} \tag{6.7}$$

wherein $a_k^w(x) \in \mathbb{R}$ is the logit scores emitted by the neural network. Additionally, standard deviation $\sigma_k(x)$ must be positive. To satisfy this, a SoftPlus (see equation 6.8) non-linearity is applied to the output of the neural network.

$$\sigma_k(x) = log(1 + exp(\beta \times x)); \beta = 1 \tag{6.8}$$

Since mean $\mu_k(x;\theta)$ doesn't have any constraint, we used only linear layers without

any non-linearity for the respective output neurons.

6.2.1 *Objective and Losses*

Training the network has two objectives: on one side we want the decoder output to resemble the network input, as in reconstruction-based anomaly detection. This forces the encoder to catch features that are relevant to describe the normal data. On the other side, the goal is to train the Gaussian mixture density network to model the manifold where the encoded features of normal images reside. For the reconstruction-based part we adopted a combination of two losses:

- *Mean Squared Error (MSE)*: it is a pixel-level loss, which assumes independence between pixels. MSE loss is computed as the average of the squared pixel-wise differences of the two images, and can be formally defined in terms of the Frobenius norm as $\frac{1}{WH}\|X - \hat{X}\|_F^2$, where X is the input and $\hat{X}$ is the output of the decoder network (respectively the original and the reconstructed image), and W, H are the image width and height respectively.

- *Structural Similarity Index* - The Structural Similarity Index (SSIM) [13] is used to measure the image similarity by considering visual properties that are lost in the standard MSE approach:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{6.9}$$

  where, μ_x, μ_y, are the average values of input and reconstruction image, σ_x^2, σ_y^2 are the variance of input and reconstructed image, σ_{xy} is their co-variance and c_1, c_2 are the two constants used for numerical stability.

For the Gaussian mixture density network training, we used the Log-Likelihood Loss (LL). The parameter θ of the Gaussian estimation network is fitted through maximum

likelihood estimation. We minimize the negative conditional log-likelihood of the normal class training data.

$$\theta^* = -\arg\min_{\theta} \sum_{k=1}^{K} \log p_\theta(y_n|x_n) \tag{6.10}$$

For the purpose of regularization, we also add Gaussian noise $\mathcal{N}(0, 0.2)$ to the transformer embedded features before feeding it to the GMM model. Adding noise during training is seen as a form of data augmentation and regularization that biases towards smooth functions [14, 4].

Hence, the final objective function to minimize is the weighted addition of the above three losses.

$$L(X) = -LL + \lambda_1 MSE(X, \hat{X}) + \lambda_2 SSIM(X, \hat{X}) \tag{6.11}$$

wherein, $\lambda_1 = 5$ and $\lambda_2 = 0.5$, are the scalar weights found heuristically for all the datasets used in this study.

6.3 Experimental Results

In this section, we present the experimental results obtained by our proposed network VT-ADL. We first describe the used datasets, training and testing procedures, and comparative results. We also introduce the beanTech Anomaly Detection Dataset[1] (BTAD), a novel dataset of real-world, industry-oriented images composed of both normal and defective products. The defective images have been pixel-wise manually labeled with a ground-truth anomaly localization mask.

6.3.1 *Datasets and Results*

- **MNIST**: MNIST dataset consists of 60K gray images of handwritten digits. Although this dataset was not originally developed for anomaly detection tasks, it

[1] http://avires.dimi.uniud.it/papers/btad/btad.zip

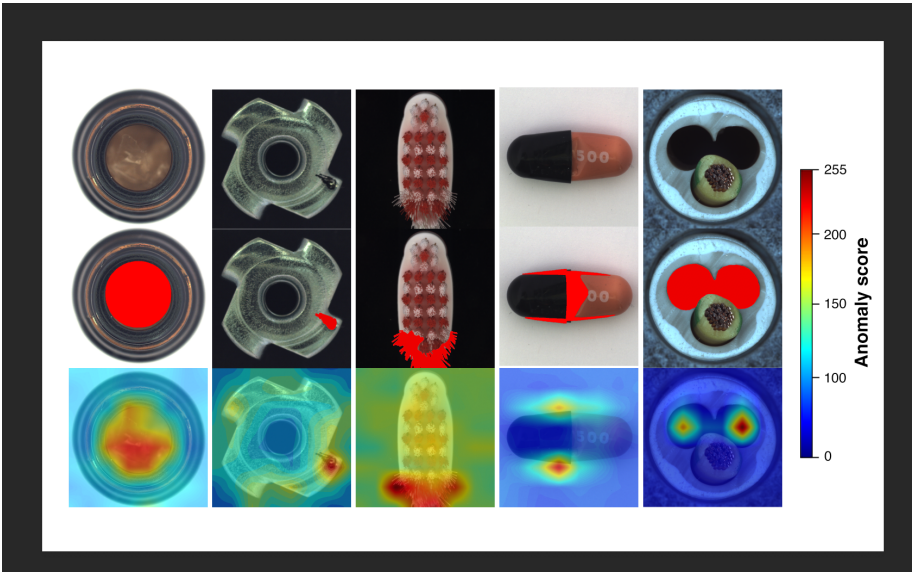


Figure 6.4: Anomaly detection on MVTec dataset. The first row shows the actual anomalous image of bottle, cable, capsule, metal nut, and brush. The second row shows the actual ground truth and the third row shows the generated anomaly score and anomaly localization by our method

has often been used as a baseline dataset, thus we used it to compare with other state-of-the-art approaches. For training, one class has been considered as normal, while all others as anomalies.

- **MVTec Dataset**: It's a real-world anomaly detection dataset. It contains 5,354 high-resolution color images of different textures and objects categories. It has normal and anomalous images which showcase 70 different types of anomalies of different real-world products. It contains grayscale images as well as RGB images. Grayscale images are quite common in industrial scenarios. With this dataset, all the images were first scales to 550×550 pixels and then center cropped to 512×512pixels before being passed to the model.

- **BTAD Dataset**: It contains RGB images of three industrial products (see Fig 6.5). Product 1 is 1600×1600 pixels, product 2 is 600×600 and product 3 is

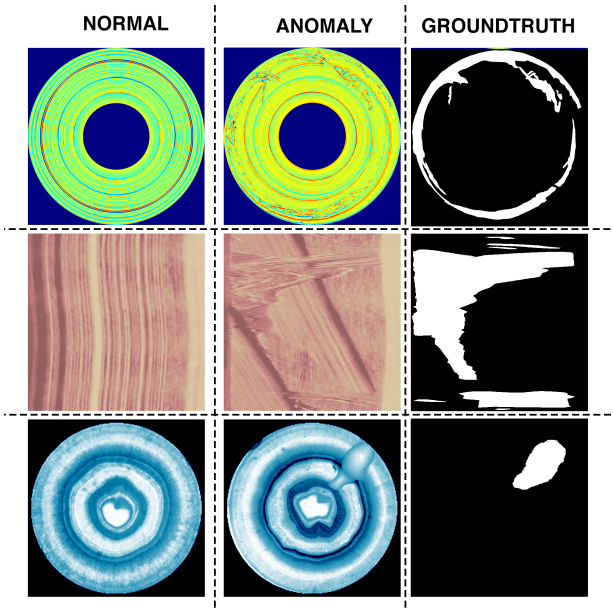


Figure 6.5: BTAD dataset. First column: normal images pf thee industrial products; Second column: anomalous images; Third column: pixel-precise ground truth

800×600 pixels in size. Product 1, 2, and 3 have 400, 1000, and 399 train images respectively. For training all the images were first scaled to 512 before passing to the model. For each anomalous image, a pixel-wise ground truth mask is given.

For training, we fed our model using the normal class data only. And for testing, a combination of reconstruction losses and the maximum of the log-likelihood loss are used to perform global anomaly detection, while the log-likelihood loss alone is used for anomaly localization. In this second case, we stored the log-likelihood loss for all the patch positions and then upsample it using 2D bilinear-upsampling, to input image size, to obtain the heatmap. Then we employed the PRO (Per Region Overlap) [12, 11] as the evaluation metric for the MVTec and BTAD datasets. For computing PRO, heatmaps are first thresholded at a given anomaly score to make the binary decision for each pixel. Then the percentage of overlap with the ground truth (GT) is computed. We followed

the same approach as in [12], to find the PRO value for a large number of increasing thresholds until an average per-pixel positive rate of 30% is reached. For the MNIST dataset, we adopted AUC (area under ROC curve) as a performance metric in order to show comparative results (see table 6.2).

The hyper-parameters used in the training are shown in table 6.1.

Adams lr rate	0.0001
Weight decay	0.0001
Batch Size	8
Epochs	400
No. of Gaussian's	150
Patch Dimension	P = 64

Table 6.1: Training hyperparameters

Class	**OC SVM**	**KDE**	**DAE**	**VAE**	**Pix CNN GAN**	**LSA**	**Deep SVDD**	**Pyr. AE**	**VT-ADL**
0	0.988	0.885	0.991	0.994	0.531	0.993	0.98	0.995	**0.99**
1	0.999	0.996	0.999	0.999	0.995	0.999	0.997	0.999	**1**
2	0.902	0.71	0.89	0.96	0.478	0.959	0.917	0.941	**0.976**
3	0.950	0.693	0.935	0.947	0.517	0.966	0.919	0.966	**0.976**
4	0.955	0.844	0.921	0.965	0.739	0.956	0.949	0.960	**0.984**
5	0.968	0.776	0.937	0.963	0.542	0.964	0.885	**0.972**	0.971
6	0.978	0.861	0.981	0.995	0.592	0.994	0.983	0.993	**0.995**
7	0.965	0.884	0.964	0.974	0.789	0.980	0.946	**0.993**	0.99
8	0.853	0.669	0.841	0.905	0.340	0.953	0.939	0.895	**0.974**
9	0.995	0.825	0.96	0.978	0.662	0.981	0.965	0.989	**0.99**
Mean	*0.95*	0.81	0.94	0.97	0.62	0.97	0.948	0.97	**0.984**

Table 6.2: AUC results of anomaly classification using MNIST, Each row shows the normal class of the trained model. Comparative results are taken from [1, 67]

Table 6.3 shows the results for the MVTec dataset. The value shows the PRO curve up to an average false positive rate per pixel of 30% is reported. It measures the average overlap of each ground truth region with the predicted anomaly region for multiple thresholds. Our proposed methods performed at par with the most recent state-of-the-art algorithms (results taken from [12]) and even outperformed them in 7 product categories. For our newly published BTAD dataset, we are also reporting the

Category	**1-NN**	**OC SVM**	**K Means**	**AE MSE**	**VAE**	**AE SSIM**	**Ano GAN**	**CNN Feat. Dic**	**Uni. Stud.**	**VT-ADL (Ours)**
Carpet	0.512	0.355	0.253	0.456	0.501	0.647	0.204	0.469	0.695	**0.773**
Grid	0.228	0.125	0.107	0.582	0.224	0.849	0.226	0.183	0.819	**0.871**
Leather	0.446	0.306	0.308	0.819	0.635	0.561	0.378	0.641	**0.819**	0.728
Tile	0.822	0.722	0.779	0.897	0.87	0.175	0.177	0.797	**0.912**	0.796
Wood	0.502	0.336	0.411	0.727	0.628	0.605	0.386	0.621	0.725	**0.781**
Bottle	0.898	0.85	0.495	0.91	0.897	0.834	0.62	0.742	0.918	**0.949**
Cable	0.806	0.431	0.513	0.825	0.654	0.478	0.383	0.558	**0.865**	0.776
Capsule	0.631	0.554	0.387	0.862	0.526	0.86	0.306	0.306	**0.916**	0.672
Hazelnut	0.861	0.616	0.698	0.917	0.878	0.916	0.698	0.844	**0.937**	0.897
Metal Nut	0.705	0.319	0.351	0.83	0.576	0.603	0.32	0.358	**0.895**	0.726
Pill	0.725	0.544	0.514	0.893	0.769	0.83	0.776	0.46	**0.935**	0.705
Screw	0.604	0.644	0.55	0.754	0.559	0.887	0.466	0.277	**0.928**	**0.928**
Toothbrush	0.675	0.538	0.337	0.822	0.693	0.784	0.749	0.151	0.863	**0.901**
Transistor	0.68	0.496	0.399	0.728	0.626	0.725	0.549	0.628	0.701	**0.796**
Zipper	0.512	0.355	0.253	0.839	0.549	0.665	0.467	0.703	**0.933**	0.808
Means	0.64	0.479	0.423	0.79	0.639	0.694	0.443	0.515	0.857	0.807

Table 6.3: Comparative results on the MVTec dataset. Comparative results are taken from [12].

Prdt	**PRO Score ours**	**PR AUC ours**	AE MSE	AE MSE+SSIM
0	*0.92*	0.99	0.49	0.53
1	*0.89*	0.94	0.92	0.96
2	*0.86*	0.77	0.95	0.89
Mean	*0.89*	**0.90**	0.78	0.79

Table 6.4: Results on BTAD dataset. We also compare our PR-AUC with the results of convolutional autoencoders trained with MSE loss and MSE+SSIM loss.

first results in table 6.4 with a similar model configuration as of MVTec. For comparison, we also report the PR-AUC of a basic convolutional autoencoder on BTAD with MSE and MSE+SSIM loss.

6.3.2 *Ablation Studies*

Here we discuss our ablation studies and tweaked methods which justify our choice of network and show the interesting configurations Ano-VT can be used in different anomaly detection scenarios. Most of the ablation studies are done on select products

from the MVTec dataset.

Gaussian mixture model tuning

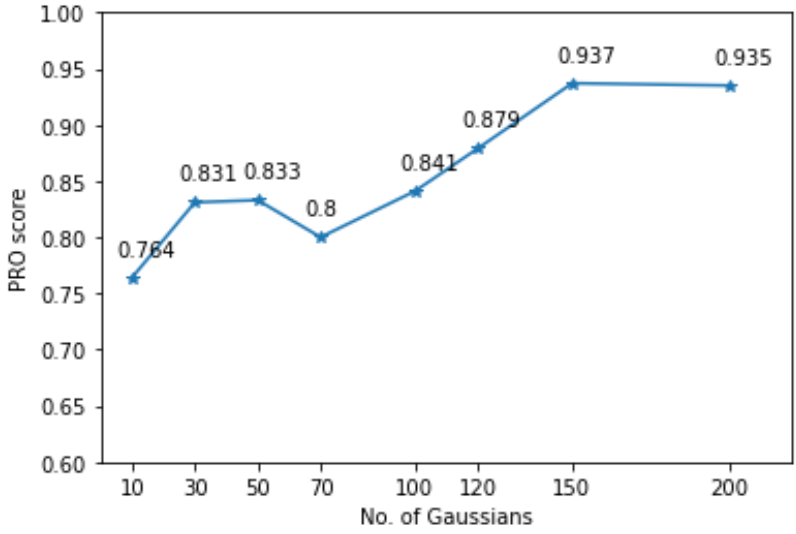


Figure 6.6: Plot shows the PRO score for the different no of Gaussians used in the Gaussian approximation.

Here we justify the choice of a number of Gaussians for our mixture model. For this, we trained on the MVTec dataset with an increasing number of Gaussians and calculated the PRO-score (Fig.6.6). we found that with an increasing number of Gaussians, PRO-score increases and then becomes constant. We also tried to see the effect of noise addition in the transformer encoded features for generalization. With noise added, the PRO score with 150 Gaussians is *0.897* in contrast to *0.807* without noise. Hence, noise addition actually helps in generalizing the learning procedure.

Inductive Bias Induces Average Learning

We also tried to check the effect of the dynamic routing algorithm on the results. We found that if we don't use the reconstruction vector, (see section 6.2), the network tries

to average out the learning. A training reconstruction can be seen in the Figure6.7. When we use the reconstruction vector(projected via projection matrix) the network becomes equivariant and captures the different orientations of the same product in the reconstruction. This is a problem of inductive bias [33] in vision transformer. However, this problem can be solved using the projection matrix, which forces the network to become equivariant. Additionally, projection matrix also helps in capturing better causal features at the transformer encoder.

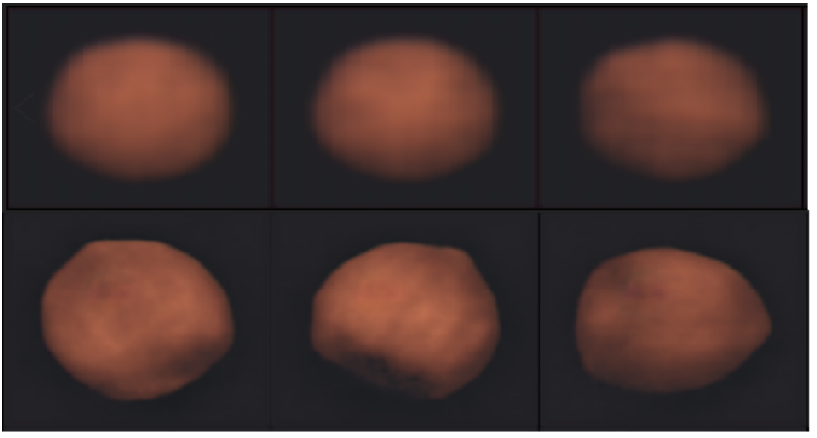


Figure 6.7: First row: Reconstructions of average learning without reconstruction vector; Second Row: Reconstructions of Learning with reconstruction vector

7

Industrial Application: How to choose a model for Image Anomaly Detection and Localization

> "If you have a lot of data and you want to create value from that data, one of the things you might consider is building up an AI team."
>
> – *Andrew Ng*

Localizing the image anomalies is predominantly perceived as the task of pixel-level image segmentation. The fact that industrial datasets are highly imbalanced (a high amount of normal image data is available), and industries deterrence to invest

in data annotation has attracted far less research interest than image-level anomaly detection. There are several attempts made to the anomaly detection task using the unsupervised approach for image-level classification, but no relevant attempts have been done for anomaly localization. The prime reason is the non-availability of pixel-precise annotated data or limited absorption of advanced AI technologies at an industrial scale [104]. In addition to this, the legacy system used by industries and the minimalist approach to upgrade present IT infrastructure in many industries also deters them from using the recent and state-of-the-art AI Visual Inspection Systems (VIS). Due to the increasing complexity of machines, downtime of any kind can affect the overall success of a company. That's why companies are looking for new ways to manage this issue cost-effectively [104, 39].

Lately, we find that some encouraging attempts have been done in this limiting field, where a single network has been developed, which can do the image anomaly detection and localization at the same time [1, 70, 93]. The most interesting thing about such works is that they can be trained in an unsupervised fashion, which means, they don't need pixel-precise ground truth for the training, and they are really fast in their inference time, while they can easily be integrated with any of the present IT infrastructures.

Hence, this comparative study seeks to provide a structured and comprehensive overview of the research carried out on possible Deep Anomaly Detection (DAD) solutions in real-life applications in industries. The selection of appropriate machine learning techniques in the field of the manufacturing industry is challenging, as numerous aspects have to be considered before adoption. This study considers some previous state-of-the-art segmentation networks, which are scarcely used in the image anomaly detection task, contrary to the new patch-based networks. These new patch-based techniques possess an interesting situation as they don't need any ground truth for the pixel-level segmentation. Hence, a single network is capable of both image-level classification and pixel-level classification.

We studied some of the most recent approaches developed in recent times like SPADE [27], and VT-ADL [70] and compared them with the segmentation networks like FCN32 [106], Unet [95], Unet++ [131], UNet2 [48], SegNet [8], used for the same VIS purpose and we found some underlying conclusions from this comparative study. The comparison will mainly include the *Network complexity*, *Training complexity*, *Inference time*, *IT infra requirements*, *Accuracy of the network*, and *Agile nature* for future adaptation.

7.1 Deep Learning Models

We use some of the most successful segmentation networks, in contrast with the very recent patch-based models. As posed in their respective published work, recent methods are showing an edge over the previous state-of-the-art methods. Hence, it's really interesting from an industrial application point of view to critically study these methods. For the purpose of this study we performed comparative evaluations on the following anomaly detection techniques:

- **FCN32** [1] [106]: It is one of the most popular Fully Convolutional networks for image segmentation. In this approach, an image is downsized passing through the convolutional layers. The output of the convolutional layer is a feature of size smaller than the input image. The output is then up-sample using transposed convolution layers[62] to get the pixel-wise, image size output (label map). The model was an initial attempt at the image segmentation task and produced good results on various types of datasets. We use this model for comparison in order to test the performance of such benchmarking models in contrast to the recent attempts using patch-based methods for anomaly localization tasks.

- **Unet**/ **Unet**++ / **Unet2**[2][3] [95, 131, 48]: It's an U-shaped convolutional network

[1] https://github.com/pochih/FCN-pytorch
[2] https://github.com/4uiiurz1/pytorch-nested-unet
[3] https://github.com/upashu1/Pytorch-UNet-2

of a specific encoder-decoder scheme: The encoder squeezes the spatial dimension while increasing the number of channels. On the other hand, the decoder network just does the opposite of the encoder in each successive layer. The input to the decoder is called a "Bottleneck". Then there are skip connections directly between the encoder and decoder network, which passes the high-level feature maps, which helps in getting rich features at each layer of the decoder. In the end, the decoder restores the image dimension to make predictions for each pixel in the input image. These kinds of models are highly applicable in real-life scenarios. In fact, most of the industries which are using the deep learning solution for any kind of Visual Inspection Systems are using one of these adapted networks. Unet++ and Unet2 are the new variants of the original Unet network, with optimized operations and lower parameters. This allows for faster training and faster inferences. Tiny Unet [4] is the vanilla Unet with half the layer of the vanilla Unet, this has been especially adapted for faster industrial applications.

- **SegNet** [5] [8]: SegNet is also an encoder-decoder scheme-based network followed by a final pixel-wise classification layer The encoder is usually a pre-trained classification network like VGG [56] or ResNet [41] or MobileNet[44]. The major difference with the above-mentioned networks includes a) similar to upsampling approach it uses an approach called *Unpooling*, b) it doesn't use pooling indices rather it transfers the entire feature maps from the encoder to the decoder, then with the concatenation to perform convolution. These two approaches make the model larger and memory hungry.

- **SPADE** [6] [27]: Semantic Pyramid Anomaly Detection doesn't require any training, rather it uses a pre-trained deep neural network to extract image features,

[4] https://ngc.nvidia.com/catalog/resources/nvidia:unet_industrial_for_tensorflow
[5] https://github.com/delta-onera/segnet_pytorch
[6] https://github.com/byungjae89/SPADE-pytorch

then it arranges the features of the normal images using nearest K normal images. At last, it finds the pixel-level correspondence between the target image and the normal image. The novel target image which doesn't match the retrieved normal images is labeled as anomalous.

- **VT-ADL** [7] [70]: Vision Transformer for Image Anomaly Detection and Localization is a patch-based approach that uses a vision transformer network to encoded the patches of the normal images and tries to learn the distribution of these patches using a Gaussian Mixture Network[14]. It labels a novel target image as anomalous based on higher log-likelihood loss and the patch-based losses are used for the anomaly localization.

7.2 Techniques used for the comparison

First, a direct comparison has been made by training each of the models separately and testing it on MVTec [11] and BTAD [70] datasets. We used Per Region Overlap (PRO) score [12, 70] for the comparison.

Additionally, we tried to compare various other meta-features of the models for example *training complexity, size of the network, inference time, and agile nature*. Below is the explanation of what all we covered in the meta feature analysis.

- **Training Complexity**: Even though deep learning showed remarkable success, it struggles to find a preferential position in real-life industrial scenarios. Deep networks have proven their prowess in processing and finding patterns using big data, where traditional machine learning algorithms still don't perform efficiently. The price of this remarkable success of deep learning is highly dependent on the resources and hence, it's important to know what we are trading off while we are

[7] https://github.com/pankajmishra000/VT-ADL

choosing a deep learning approach to solve a real-life industrial scenario. In this research we tried some of the most important factors which define the training complexity of a model, we compare practical factors like (see table 7.1) -

- ***Total parameters of the model***(see Fig7.1, shows the total trainable parameters used in the model. Ideally, we desire lesser parameters for efficient learning, else the model becomes complex.
- ***Multi-stage training*** requires, training a network in two or more than two stages. usually, such training procedures are complex and unstable, for example, GAN's [38].
- ***Pre-trained networks*** are used in various methods to extract the features from images. Usually, the parameters of these networks are not updated during the backward pass of an overall training procedure.
- ***Multiple loss functions*** are used in various deep learning approaches, as they force a network to learn a particular regularity while training. Interestingly, a higher number of loss functions results in complex training procedures.
- ***Ground truth/Annotated data*** are required for supervised learning procedures. This could be a limiting aspect of a deep learning approach, as annotating data can be time-consuming and costly.
- ***Training Procedure*** plays an important role in deep learning solutions. Supervised learning can be very accurate but requires lot of annotated and balanced datasets. On the other hand semi-supervised or unsupervised learning saves us from the requirement of annotated data.
- ***Training with high imbalanced datasets*** are important in critically analyzing the models when comparison needs to be for industrial scenarios. In industries, highly imbalanced datasets are quite mundane and industries

have limited interest in annotating them. Hence, a model capacity to get trained on such datasets. while at the same time not compromising with the performance is an ideal candidate for the industrial use cases.

- ***Minimum data needed*** for deep learning methods shows the learning and generalizing capacity of a network. Lower data-hungry networks are always preferred over the higher data-hungry networks.
- ***Training time for the first 100 epochs*** are measured in order to check how fast a network can be trained.
- ***Presence of attention Module*** in a network helps in learning the important features from the dataset. Attention networks have already shown their superiority in the language and vision field [116, 33].
- ***Ability to work with high-definition images*** is a particularly desirable property for a deep learning network opted for an industrial use case. This is because high-definition images are common in industrial VIS systems. Also, higher image pixels demand higher computing resources, which is an important factor from a cost point of view in industries.

- **Size of Network**: Size of the network is a very important factor when we talk in the context of real-life deployment. This is especially keeping the fact that most of the industries try to deploy the AI solutions either on their present IT infrastructure or they want it to be deployed at some edge devices. In both cases, size plays an important matter. Hence, we tried to study the size of the deep models (see section 7.1) in the forward pass, backward pass, full model size, and ONNX [8] exported model size, see figure 7.2.

- **Inference Time**: Inference time is an important parameter to compare as it decides if the network can be deployed for real-time deployment. Hence, we tried

[8] https://onnx.ai/

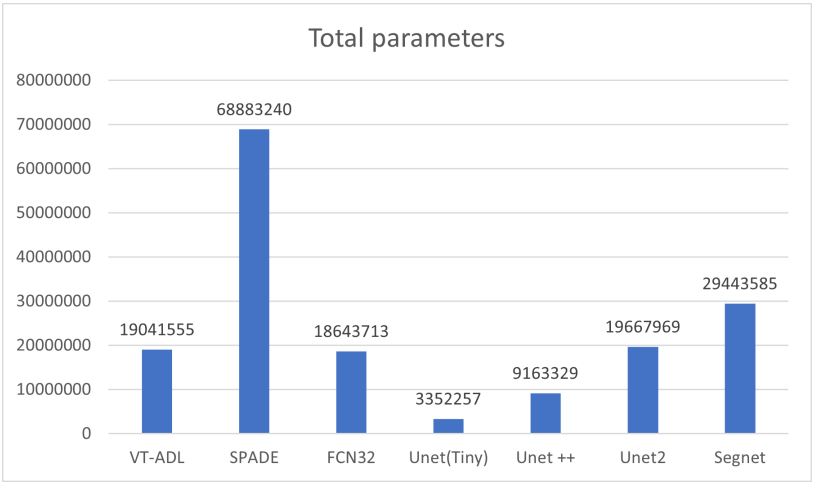


Figure 7.1: Total trainable parameters of the deep learning model used for this study.

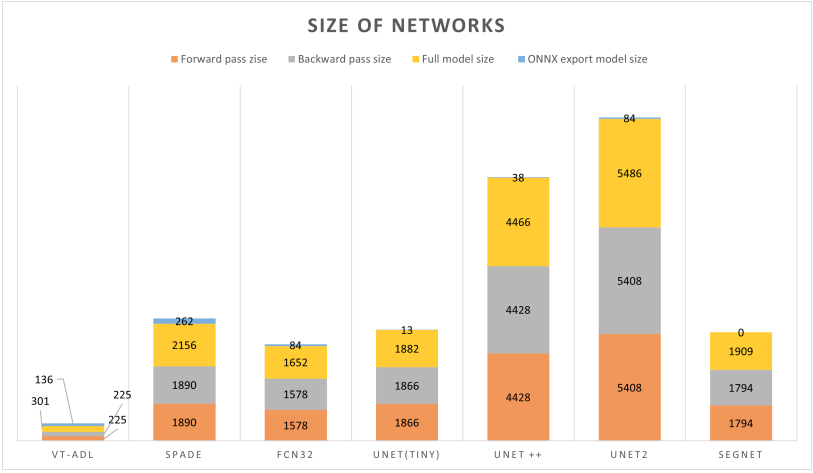


Figure 7.2: Network Sizes: The stacked bar plot shows the size (in MB) of the forward pass, backward pass, full size model and the ONNX-exported model size of the deep models used in this study.

to compare the inference time of the single precision model(32 bit)(see figure 7.3 and half precision model(16 bit) over GPU (Nvidia Tesla K40 11GB) and single precision model over CPU (CPU - i5 3,1GHz) (see Figure 7.4 and table 7.2).

Table 7.1: Network training complexity of the deep learning models

Categories	**VT-ADL**	**SPADE**	**FCN32**	**Unet(Tiny)**	**Unet ++**	**Unet2**	**Segnet**
Total parameters	19041555	68883240	18643713	3352257	9163329	19667969	29443585
Multi-stage training	No	No	No	No	No	No	No
Pre-trained network	No	Yes	No	No	No	No	No
Multiple Loss function	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Ground truth required	No	No	Yes	Yes	Yes	Yes	Yes
Supervised/ semi-supervised	semi-sup	semi-sup	sup	sup	sup	sup	sup
imbalanced dataset training	Yes	Yes	No	No	No	No	No
Min.number of Image Data needed	100	100	200-400	200-400	200-400	200-400	200-400
Training time needed (sec.) /100ephs	734.92	255.29	1488.52	1316.11	1961.15	1886.41	1709.16
Attention module	Yes	No	No	No	No	No	No
work on Patches	Yes	Yes	No	No	No	No	No

Table 7.2: Inference time(in sec.) of the deep models measured over GPU and CPU.

	VT-ADL	**SPADE**	**FCN32**	**Unet(Tiny)**	**Unet ++**	**Unet2**	**Segnet**
single precision	**0.01**	0.035	0.037	0.021	0.059	0.057	0.044
half precision	**0.0083**	0.033	0.028	0.016	0.055	0.04	0.042
full precision	**0.03**	1.75	2.81	1.45	6.29	6.13	5.3
Global AD/ Anomaly Localization	**both**	**both**	local	local	local	local	local

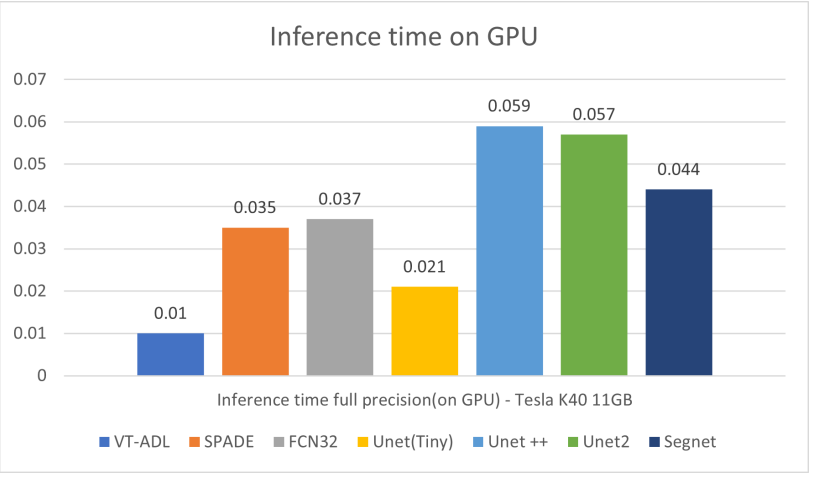


Figure 7.3: Inference time (in sec) of full precision models over GPU from all the deep models.

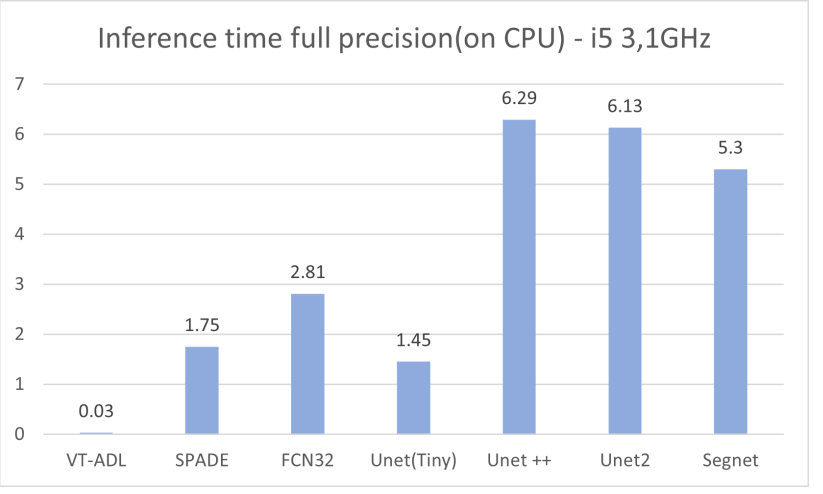


Figure 7.4: Inference time(in sec) of full precision models over CPU from all the deep models.

7.3 Experimental Results

In this section, we show the comparison results obtained by training the deep models over the MVTec (see table 7.3) and BTAD datasets (see table 7.4). We computed the

PRO score for all the models. A global level image classification score i.e. a Precision recall curve is not employed in this comparison study.

7.3.1 *Datasets and Results*

In literature various methods were tested on datasets that were not suited for anomaly detection task (eg. MNIST [60] and CIFAR [56]), and only recently few datasets have been released specifically for anomaly detection, with a focus on industrial products. We used the following dataset for this study:

- **MVTec Dataset** [11]: It's an industrial anomaly detection dataset. It contains 5,354 high-resolution color images of different textures and objects categories, see figure 4.3. It has normal and anomalous images which showcase 70 different types of anomalies of different real-world products. It contains grayscale images as well as RGB images. Grayscale images are quite common in industrial scenarios. With this dataset, all the images were first scaled to 550×550 pixels and then center cropped to 512×512 pixels before being passed to the model.

- **BTAD Dataset** [70]: It contains 3 industrial products both in RGB colour spectrum. Product 1 is 1600×1600 pixels, product 2 is 600×600 and product 3 is 800×600 pixels in size, see fig 6.5. Product 1, 2, and 3 have 400, 1000, and 399 train images respectively. While training all the images were first scaled to 512 before passing to the model.

7.4 Discussion

In this section, we discuss the results of the comparative techniques used and the classification results.

Table 7.3: Inference performance of the models. the table shows the PRO scores of the model over the MVTec dataset. The best PRO score in the category has been highlighted in bold.

Product	**SPADE**	**FCN32**	**Unet(tiny)**	**UNet2**	**Unet++**	**SegNet**	**VT-ADL**
bottle	0.94	0.96	**0.97**	**0.97**	**0.97**	0.94	0.77
cable	0.84	0.87	0.91	0.96	0.94	0.83	0.87
capsule	**0.98**	0.57	0.77	0.56	0.72	0.59	0.73
grid	**0.99**	0.79	0.90	0.93	0.89	0.90	0.78
hazelnut	**0.99**	0.97	0.98	0.98	0.98	0.97	0.95
pill	0.95	0.95	**0.99**	0.97	**0.99**	0.93	0.67
metal_ nut	0.95	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	0.90
screw	**0.99**	0.63	0.78	0.66	0.74	0.54	0.73
tile	0.87	0.98	0.98	**0.99**	0.98	**0.99**	0.71
toothbrush	**0.99**	0.88	0.97	0.97	0.98	0.90	0.93
transistor	0.76	0.90	0.96	**0.97**	0.96	0.93	0.90
wood	0.96	0.90	0.96	**0.98**	0.97	0.95	0.80
zipper	**0.99**	0.78	0.87	0.93	0.91	0.83	0.81
Means	**0.94**	0.86	0.92	0.91	0.93	0.88	0.81

Table 7.4: Inference performance on the BTAD dataset. Best PRO score in each category has been highlighted in bold

Product	**SPADE**	**FCN32**	**Unet(tiny)**	**UNet2**	**Unet++**	**SegNet**	**VT-ADL**
Mech part body (0)	**0.98**	0.92	0.93	0.97	0.91	0.94	0.92
Mech part surface (1)	**0.98**	0.87	0.91	0.93	0.91	0.90	0.89
Motor case (2)	**0.99**	0.87	0.95	0.97	0.95	0.84	0.86
Mean	**0.98**	0.89	0.93	0.96	0.92	0.89	0.89

First, we discuss the results of the comparative techniques used. We are comparing the newly suggested methods like VT-ADL and SPADE against the other segmentation networks. Starting with training complexity (see figure 7.1) - Total trainable parameters are important for any deep model and it also defines the complexity and size of the network. We find that networks like Tiny Unet followed by Unet++ have the least trainable parameters, while the Segnet has 1.5 times (approx) more trainable parameters in contrast to VT-ADL. While VT-ADL and Unet2 are at par with the trainable parameters. Although SPADE has the highest number of parameters, they are not trainable, as SPADE doesn't require any training, rather these are the trained parameters of the backbone network it uses for the feature extraction. Hence, considering trainable parameters Tiny Unet has edge over other deep models.

Other training complexities of the networks can be seen in Table7.1. The most important of them is a type of training procedure and the ability to train the network on high-imbalance datasets. All the segmentation networks are trained in the supervised fashion, i.e. they need labeled data, in this case, pixel-precise ground truth mask for the segmentation. On the contrary VT-ADL and SPADE, trained in semi-supervised/unsupervised fashion, don't need any masks for the training. Hence, this gives the new approaches a competitive edge over the segmentation networks. Additionally, VT-ADL has an attention module, which helps in exploiting the benefits of self-attention while training. Encouragingly, VT-ADL and SPADE, both can work on image patches, while other segmentation networks don't have these abilities. Working on image patches is indeed an advantage as many industrial image acquisition systems produce high/ultra-high-definition images, which are tough to processes with limited computing resources.

Comparing the size of the network, we find that Tiny Unet is again the smallest when we see only the exported ONNX model. While if we combine the forward pass size, backward pass size, and full model size, VT-ADL emerges the smallest overall (see

figure7.2). This factor is important as the size of the network forward and backward pass determines the batch size while training a network. While ONNX model size is helpful in determining the resource requirements during real-time deployment. For this study, we were not able to export Segnet to ONNX, because of the technical limitations of ONNX library.

Inference time plays crucial role when we talk about the real-time deployment of the deep models, hence we tried to compare the inference performance and inference time of the models. Inference performance can be seen in table 7.3 and table 7.4. We can see that SPADE performed best in all the networks in most of the categories with a close margin to Unet++. Inference time can be seen in Figure 7.3 over GPU and Figures 7.4 over CPU. Interestingly inference time of SPADE is 0.035 sec (on GPU) and 1.75 sec (on CPU). The lowest inference time is of VT-ADL (0.01 sec over GPU and 0.03 sec over CPU). VT-ADL has also performed at par with most of the segmentation networks with the lowest inference time, both on GPU and CPU. Interestingly Tiny Unit has also shown at par inference performance with SPADE and UNet++ and second-lowest inference time over GPU in case of single precision and half precision, while it lagged on CPU. The reason for lowest inference time for VT-ADL is that it doesn't have any convolutional operations, hence, it has very fast executions.

This chapter shows a comparative study between some of the widely used deep network for the segmentation task and the recently developed approach like VT-ADL. We have shown that how the recent network are more industry friendly and can be easily used in the real life industrial scenarios. In the next chapter we explored the opportunities given by "Continual Learning" and "Few Shot Learning" in the field of DAD. And how the future of industrial scale anomaly detection is lying at the interjection of these two approaches.

8

Future Work

"Without the capability of retaining and accumulating knowledge learned in the past, making inferences about it, and using the knowledge to help future learning and problem solving, achieving artificial general intelligence (AGI) is unlikely.."

– Zhiyuan Chen and Bing Liu, Lifelong Machine Learning

After the take-off of *Deep Learning* (DL) [59], especially after 2012 and the following groundbreaking work by other researchers, has open the path to a broader range of applications, whose complexity was even unthinkable to tackle a few decades ago. In fact, recent novel learning algorithms, like their biological counterparts, would likely access huge volumes of high-dimensional, multi-domain, real-time data from complex and constantly evolving environments in order to scale in terms of intelligence [50] and adapt to the evolving circumstances continually over time.

If we see the application of deep learning in the anomaly detection task then the above sentences are apt and are also a reason for motivation to see continuous learning (CL) as the solution to the highly evolving industrial needs. In all of the previous chapters, we talked about the supervised or unsupervised method to tackle the anomaly detection task. All these are trained once and are then used for inference. These methods performed well in industrial scenarios, they lack the ability to improve over time. Industries these days are more interested in the methods which automatically adapted themselves to the new products, hence, reducing the downtime to a minimum. Additionally, the problem of *Catastrophic Forgetting* [54], makes them unfit for sequential learning. Hence, to provide the solution to the newly evolving industrial anomaly detection task, continual learning methods are posing promising solutions.

8.1 Continual Learning for Anomaly Detection

With the recent progress in CL and its applications, there are three fuzzy categorizations of the most common strategies [30]:

- **Replay Methods**: These methods tend to store the samples in raw format or generate a similar-looking sample with generative models. The previously learned samples are replayed while learning a new task to avoid forgetting. These methods reuse a model input for rehearsal, or for constrained optimization to prevent the previously learned tasks. The most common examples of these methods are - SER [47], TEM [23], DGR [108], LGM [91], GEM [63] etc.
- **Regularization Based Methods**: These methods usually avoid storing raw data, prioritizing privacy, and thus reducing memory requirements. Here an extra regularization factor is introduced in the loss function which helps in retaining the previously learned weights while learning on new data. Some of the major work in this field are - EWC [54], IMM [61], DMC [128], MAS [3], EBLL, [92] etc.

- **Parameter Isolation Methods**: These methods are based on an idea to have a separate model for separate work. So, these models dedicate different model parameters to each of the new tasks in order to prevent possible forgetting. While these approaches can have a huge memory requirement and can also grow new branches making the network architecture too huge to manage. Various attempts have been made to make the network part static with fixed parts allocated to dedicated work. Some of the major examples are - packet [65], PathNet [36], HAT [105], PNN [98], RCL [126], DAN [96] etc.

In all the approaches, *regularisation-based methods show the most promising solutions* for the deep anomaly detection (DAD) task up to industrial levels. The methods like EWC and IMM are suggesting for new mathematical ways to tie the newly learned weights over the new task with the previously learned task, without increasing the network size. Hence, these approaches can be used to transform the methods discussed in the previous research into CL methods for anomaly detection. Hence, the future of anomaly detection work lies in the exploration of Continual Learning approaches.

8.2 Few Shot Learning for Anomaly Detection

Like CL, Few Shot Learning (FSL) is a newly explored branch of deep learning. The need to train the data-hungry deep neural network with fewer data is in high demand, especially in industrial use cases. Hence, exploring the paradigms of FSL for the anomaly detection task is very encouraging to solve the anomaly detection problem at an industrial scale. FSL can rapidly generalize to new tasks containing only a few samples with supervised information. The recent developments like Proto-typical nets [110], Matching Networks [119], PMN [121], SNAIL [66], DCCN [129], and TADAM [74] provide various kinds of embedding learning with fewer datasets.

As discussed in the previous chapters major problem of deploying DL approaches for

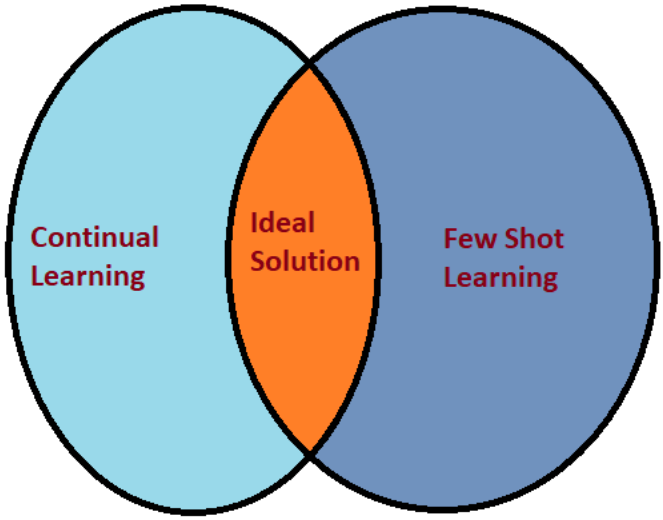


Figure 8.1: Image showing that ideal solution will lie at the interaction of CL and FSL

the anomaly detection task at the industrial application is the fewer or high imbalanced availability of the datasets. Additionally, the cost of data annotation is very high. Hence, if to minimize the cost of data annotation which in turn means using fewer data for the training, FSL could plays an interesting role. There have been some recent approaches to try FSL approaches for anomaly detection tasks [93], but with limited success. Although most of the approaches developed are supervised learning, but that's not the limiting case in industrial scenarios. Because industries are generating huge amount of unbalanced dataset. So, using the concepts of FCL over a scare annotated dataset from both the classes could be a possible solution. Nonetheless, this field posses huge potential for further future research in the field of unsupervised or semi-supervised approaches as well.

In fact, the best solution will lie somewhere at the intersection of CL and FSL. Hence, exploring the approaches of CL combined with FSL will give the most advanced

solution to the most advanced evolving problems of the industrial needs.

9

Conclusion

"Let your concern (or focus) be on your action, let it not be on the outcome of the action. Do not act only out of expectation of a result, but then do not slip into inactivity."

– 2.47, Srimad Bhagwat Gita, Sri Krishna

Deep learning has successfully demonstrated to operate in rather vertical and self-contained context, their application is more natural, ever evolving, multi-modal, and multi-tasking has been relatively modest. The goal of deep anomaly detection (DAD) approaches explored in this dissertation is to provide novel and viable solution to the anomaly detection task in the industrial context.

Most deep anomaly methods focus on point anomalies (see chapter 1), and have demonstrated exceptionally good results than traditional methods. However, deep models have recently forayed into the high-dimensional field of group/conditional anomalies (see chapter 1), and are new challenging research topic in the anomaly detec-

tion domain. With this work we explored novel methods which are supervised, semi-supervised/unsupervised, keeping industrial needs at sight.

9.1 Supervised Global Anomaly Classification

Our supervised work with adapted capsule network (see chapter 3), solved the problem of high-imbalance data training in real-life cases. Our proposed method currently outperforms or it is comparable to other deep learning anomaly detection techniques as the ones discussed in Chapter 3, however a direct comparison would be unfair since most of those methods use semi-supervised or unsupervised techniques. We proposed an alternative approach based on fully supervised learning with imbalanced datasets. This idea came from real-world scenarios, in which anomalous data are often available but their amount is extremely scarce. The proposed approach, which is a variant of the the capsnet architecture, showed good performances even with extremely imbalanced datasets, outperforming both the standard capsnet architecture and other anomaly detection techniques.

9.2 Semi-supervised Approaches for Global Image Classification

In addition to supervised approach, see chapter 3, we also developed novel semi-supervised approaches for the global image anomaly classification. We proposed deep pyramidal network with stacked autoencoders for anomaly detection, see chapter 4. Anomalies are identified by means of a network that encodes normal images in a low-dimensional latent space and then reconstructs them, ideally modeling an identity function. Since the network is trained on normal data only, its fails at reconstructing anomalous images, which can be detected by an image similarity loss. The main contributions of this work

consist in the usage of a multi-scale pyramidal approach that extract latent features at different resolutions, and the usage of a high-level perceptual loss to better compare images at feature level, rather than at pixel level. Moreover, differing from many works that have been evaluated on basic datasets only such as MNIST, we tested the proposed network on MVTec, a real-world dataset of defective products. Achieved results are promising and often outperform other state-of-the-art methods

While in Chapter 5, we proposed a novel network PIADE, a deep reconstruction-based pyramidal approach, in which image features are extracted at different scale levels to better catch the peculiarities that could help to discriminate between normal and anomalous data. The network is trained on normal data only, and it builds a "normality model" by mapping the input images in a low-dimension feature space, from which they can be correctly reconstructed. The inability of the network to reconstruct other images allows the identification of anomalies, which can be detected by their higher reconstruction error. Compared to other state-of-the-art works, the proposed models includes a pyramidal multi-scale approach to analyze image features at different scale levels, a dynamic routing layer inspired by the architecture of capsule networks, and a high-level image comparison loss. Moreover, the system has been tested not only on standard datasets such as CIFAR10 and COIL-100 (which have not been initially created for anomaly detection experiments), but also on the recently proposed MVTec dataset of anomalies in industrial images. Experimental results showed that the proposed model is at least at-par, and often outperforms other state-of-the-art works.

9.3 Unsupervised Approaches for Global Image Classification and Localization

We proposed a transformer-based framework VT-ADL, see chapter 6 which uses reconstruction and patch-based learning for image anomaly detection and localization.

The anomalies can be detected at a global level using a reconstruction-based approach, and can be localized with the application of a Gaussian mixture model applied to the encoded image patches. The achieved results are at par with or outperform other state-of-the-art techniques. We also published BTAD, a real world industrial dataset for the anomaly detection task.

Hence, this dissertation concludes that new deep learning methods has huge potential to offer for the new age industries striving to adopt advance technologies for their VIS systems. And newer DAD methods are smart, easy to implement, have better performance, agile and less resource dependent. The ideal solution can be obtained with a balanced mix of great data preparation and carefully crafted deep models.

Bibliography

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

[2] S. Akcay, A. Atapour-Abarghouei, and T.P. Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Proc. Asian Conference on Computer Vision*. Springer, 2018. to appear.

[3] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[4] G. An. The effects of adding noise during backpropagation training on a generalization performance. *Neural Computation*, 8(3):643–674, 1996.

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

[6] Fatemeh Azmandian, Ayse Yilmazer, Jennifer G Dy, Javed A Aslam, and David R Kaeli. Gpu-accelerated feature selection for outlier detection using the local kernel density ratio. In *2012 IEEE 12th International Conference on Data Mining*, pages 51–60. IEEE, 2012.

[7] Anu Maria Babu and Anju Pratap. Credit card fraud detection using deep learning. In *2020 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, pages 32–36. IEEE, 2020.

[8] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[9] Andrew Barto, Marco Mirolli, and Gianluca Baldassarre. Novelty or surprise? *Frontiers in psychology*, 4:907, 2013.

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

[11] 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 Conference on Computer Vision and Pattern Recognition*, pages 9592–9600, 2019.

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

[13] Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. In *International joint conference on computer vision, imaging and computer graphics theory and applications*, 2019.

[14] Christopher M Bishop. Mixture density networks. 1994.

[15] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[16] Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, oct 2018.

[17] Edoardo Calia and Davide D'Aprile. *Industry4.0*, pages 309–333. Springer International Publishing, Cham, 2020.

[18] Guilherme O Campos, Arthur Zimek, Jörg Sander, Ricardo JGB Campello, Barbora Micenková, Erich Schubert, Ira Assent, and Michael E Houle. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data mining and knowledge discovery*, 30(4):891–927, 2016.

[19] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

[20] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[21] Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey. *CoRR*, abs/1901.03407, 2019.

[22] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15:1–15:58, 2009.

[23] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

[24] Guan-Hao Chen, Chun-Ling Yang, and Sheng-Li Xie. Gradient-based structural similarity for image quality assessment. In *2006 International Conference on Image Processing*, pages 2929–2932. IEEE, 2006.

[25] P. Chen, S. Yang, and J. A. McCann. Distributed real-time anomaly detection in networked industrial sensing systems. *IEEE Transactions on Industrial Electronics*, 62(6):3832–3842, 2015.

[26] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

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

[28] 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.

[29] Lucas Deecke, Robert Vandermeulen, Lukas Ruff, Stephan Mandt, and Marius Kloft. Image anomaly detection with generative adversarial networks. In *European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 3–17. Springer, 2018.

[30] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021.

[31] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

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

[33] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

[34] 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.

[35] Kevin Faust, Quin Xie, Dominick Han, Kartikay Goyle, Zoya Volynskaya, Ugljesa Djuric, and Phedias Diamandis. Visualizing histopathologic deep learning classification and anomaly detection using nonlinear feature space dimensionality reduction. *BMC bioinformatics*, 19(1):1–15, 2018.

[36] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

[37] Ugo Fiore, Francesco Palmieri, Aniello Castiglione, and Alfredo De Santis. Network anomaly detection with the restricted boltzmann machine. *Neurocomput.*, 122:13–23, 2013.

[38] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[39] Matthias Groß. *Innovationen im Zeitalter der Digitalisierung*. Springer, 2017.

[40] Frank E Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, 1969.

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

[42] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[43] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010.

[44] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[45] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7132–7141, June 2018.

[46] Ferenc Huszar, Lucas Theis, Wenzhe Shi, and Andrew Cunningham. Lossy image compression with compressive autoencoders. 2020.

[47] David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[48] Libin Jiao, Lianzhi Huo, Changmiao Hu, and Ping Tang. Refined unet v2: End-to-end patch-wise network for noise-free cloud and shadow segmentation. *Remote Sensing*, 12(21):3530, 2020.

[49] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[50] Lukasz Kaiser, Aidan N Gomez, Noam Shazeer, Ashish Vaswani, Niki Parmar, Llion Jones, and Jakob Uszkoreit. One model to learn them all. *arXiv preprint arXiv:1706.05137*, 2017.

[51] Fabian Keller, Emmanuel Muller, and Klemens Bohm. Hics: High contrast subspaces for density-based outlier ranking. In *2012 IEEE 28th international conference on data engineering*, pages 1037–1048. IEEE, 2012.

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

[53] B. Ravi Kiran, Dilip Mathew Thomas, and Ranjith Parakkal. An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. *Journal of Imaging*, 4(2), 2018.

[54] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[55] Alexej Klushyn, Nutan Chen, Richard Kurle, Botond Cseke, and Patrick van der Smagt. Learning hierarchical priors in vaes. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 2866–2875. Curran Associates, Inc., 2019.

[56] Alex Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, Dept. of Comp. Sci., University of Toronto, 2009.

[57] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[58] Aleksandar Lazarevic and Vipin Kumar. Feature bagging for outlier detection. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 157–166, 2005.

[59] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

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

[61] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. *arXiv preprint arXiv:1703.08475*, 2017.

[62] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[63] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30:6467–6476, 2017.

[64] Xingjun Ma, Yuhao Niu, Lin Gu, Yisen Wang, Yitian Zhao, James Bailey, and Feng Lu. Understanding adversarial attacks on deep learning based medical image analysis systems. *Pattern Recognition*, 110:107332, 2021.

[65] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.

[66] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*, 2017.

[67] Pankaj Mishra, Claudio Piciarelli, and Gian Luca Foresti. Image anomaly detection by aggregating deep pyramidal representations, 2020.

[68] Pankaj Mishra, Claudio Piciarelli, and Gian Luca Foresti. A neural network for image anomaly detection with deep pyramidal representations and dynamic routing. *International Journal of Neural Systems*, 30(10):2050060–2050060, 2020.

[69] Pankaj Mishra, Claudio Piciarelli, and Gian Luca Foresti. Image anomaly detection by aggregating deep pyramidal representations. In *International Conference on Pattern Recognition*, pages 705–718. Springer, 2021.

[70] Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. VT-ADL: A vision transformer network for image anomaly detection and localization. In *30th IEEE/IES International Symposium on Industrial Electronics (ISIE)*, June 2021.

[71] Mutahir Nadeem, Ochaun Marshall, Sarbjit Singh, Xing Fang, and Xiaohong Yuan. Semi-supervised deep neural network for network intrusion detection. 2016.

[72] Paolo Napoletano, Flavio Piccoli, and Raimondo Schettini. Anomaly detection in nanofibrous materials by cnn-based self-similarity. *Sensors*, 18(1):209, 2018.

[73] Sameer A. Nene, Shree K. Nayar, and Hiroshi Murase. object image library (coil-100. Technical report, 1996.

[74] Boris N Oreshkin, Pau Rodriguez, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. *arXiv preprint arXiv:1805.10123*, 2018.

[75] Gintautas Palubinskas. Image similarity/distance measures: what is really behind mse and ssim? *International Journal of Image and Data Fusion*, 8(1):32–53, 2017.

[76] Guansong Pang, Longbing Cao, Ling Chen, Defu Lian, and Huan Liu. Sparse modeling-based sequential ensemble learning for effective outlier detection in high-dimensional numeric data. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[77] Guansong Pang, Longbing Cao, Ling Chen, and Huan Liu. Learning homophily couplings from non-iid data for joint feature selection and noise-resilient outlier detection. In *IJCAI*, pages 2585–2591, 2017.

[78] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. *ACM Comput. Surv.*, 54(2), March 2021.

[79] Guansong Pang, Chunhua Shen, and Anton van den Hengel. Deep anomaly detection with deviation networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 353–362, 2019.

[80] Guansong Pang, Cheng Yan, Chunhua Shen, Anton van den Hengel, and Xiao Bai. Self-trained deep ordinal regression for end-to-end video anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12173–12182, 2020.

[81] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

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

[83] Tomáš Pevnỳ. Loda: Lightweight on-line detector of anomalies. *Machine Learning*, 102(2):275–304, 2016.

[84] Claudio Piciarelli, Danilo Avola, Daniele Pannone, and Gian Luca Foresti. A vision-based system for internal pipeline inspection. *IEEE Transactions on Industrial Informatics*, 2018. early access.

[85] Claudio Piciarelli, Christian Micheloni, and Gian Luca Foresti. Trajectory-based anomalous event detection. *IEEE Transaction on Circuits and Systems for Video Technology*, 18(11):1544–1554, 2008.

[86] Claudio Piciarelli, Pankaj Mishra, and Gian Luca Foresti. Image anomaly detection with capsule networks and imbalanced datasets. In *International Conference on Image Analysis and Processing*, pages 257–267. Springer, 2019.

[87] Stanislav Pidhorskyi, Ranya Almohsen, Donald A Adjeroh, and Gianfranco Doretto. Generative probabilistic novelty detection with adversarial autoencoders. *arXiv preprint arXiv:1807.02588*, 2018.

[88] David Powers. Evaluation: From precision, recall and f-factor to roc, informedness, markedness correlation. *Mach. Learn. Technol.*, 2, 01 2008.

[89] B. Purushothama. 9 - costing and cost of quality. In B. Purushothama, editor, *Training and Development of Technical Staff in the Textile Industry*, pages 99–114. Woodhead Publishing India, 2012.

[90] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[91] Jason Ramapuram, Magda Gregorova, and Alexandros Kalousis. Lifelong generative modeling. *Neurocomputing*, 404:381–400, 2020.

[92] Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1320–1328, 2017.

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

[94] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Icml*, 2011.

[95] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[96] Amir Rosenfeld and John K Tsotsos. Incremental learning through deep adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 42(3):651–663, 2018.

[97] 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, Stockholmsmässan, Stockholm Sweden, 2018. PMLR.

[98] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[99] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866, 2017.

[100] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[101] 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.

[102] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International Conference on Information Processing in Medical Imaging*, pages 146–157. Springer, 2017.

[103] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *International conference on artificial neural networks*, pages 583–588. Springer, 1997.

[104] Klaus Schwab. *The fourth industrial revolution.* Currency, 2017.

[105] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557. PMLR, 2018.

[106] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, 2017.

[107] Wenzhe Shi, Jose Caballero, Lucas Theis, Ferenc Huszar, Andrew Aitken, Christian Ledig, and Zehan Wang. Is the deconvolution layer the same as a convolutional layer? *arXiv preprint arXiv:1609.07009*, 2016.

[108] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *arXiv preprint arXiv:1705.08690*, 2017.

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

[110] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*, 2017.

[111] Alessandra M Soares, Bruno JT Fernandes, and Carmelo JA Bastos-Filho. Structured pyramidal neural networks. *International Journal of Neural Systems*, 28(5), 2018. 1750021.

[112] Hongchao Song, Zhuqing Jiang, Aidong Men, and Bo Yang. A hybrid semi-supervised anomaly detection model for high-dimensional data. *Computational intelligence and neuroscience*, 2017, 2017.

[113] Shan Suthaharan. Support vector machine. In *Machine learning models and algorithms for big data classification*, pages 207–235. Springer, 2016.

[114] Naonori Ueda, Ryohei Nakano, Zoubin Ghahramani, and Geoffery E Hinton. Split and merge em algorithm for improving gaussian mixture density estimates. In *Neural Networks for Signal Processing VIII. Proceedings of the 1998 IEEE Signal Processing Society Workshop (Cat. No. 98TH8378)*, pages 274–283. IEEE, 1998.

[115] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems*, pages 4790–4798, 2016.

[116] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010, 2017.

[117] GS Vidya and VS Hari. Gold price prediction and modelling using deep learning techniques. In *2020 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, pages 28–31. IEEE, 2020.

[118] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful

representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

[119] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29:3630–3638, 2016.

[120] Jinjiang Wang, Yulin Ma, Laibin Zhang, Robert X Gao, and Dazhong Wu. Deep learning for smart manufacturing: Methods and applications. *Journal of manufacturing systems*, 48:144–156, 2018.

[121] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7278–7286, 2018.

[122] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003.

[123] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[124] D. Wulsin, J. Blanco, R. Mani, and B. Litt. Semi-supervised anomaly detection for eeg waveforms using deep belief nets. In *2010 Ninth International Conference on Machine Learning and Applications*, pages 436–441, 2010.

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

[126] Ju Xu and Zhanxing Zhu. Reinforced continual learning. *arXiv preprint arXiv:1805.12369*, 2018.

[127] Pengfei Yu and Xuesong Yan. Stock price prediction based on deep neural networks. *Neural Computing and Applications*, 32(6):1609–1628, 2020.

[128] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1131–1140, 2020.

[129] Fang Zhao, Jian Zhao, Shuicheng Yan, and Jiashi Feng. Dynamic conditional networks for few-shot learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–35, 2018.

[130] Chong Zhou and Randy C. Paffenroth. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 665–674, New York, NY, USA, 2017. ACM.

[131] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: Redesigning skip connections to exploit multiscale features in image segmentation. *IEEE transactions on medical imaging*, 39(6):1856–1867, 2019.