Anomaly Detection in Textured Surfaces

by

Manpreet Singh Minhas

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2019

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see *Statement of Contributions* included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Two publications have resulted from the work presented in the thesis:

1. Manpreet Singh Minhas and John Zelek "AnoNet: Weakly Supervised Anomaly Detection in Textured Surfaces" (To be submitted to IEEE Transactions on Image Processing)

2. Manpreet Singh Minhas and John Zelek "Defect Detection using Deep Learning from Minimal Annotations" (Submitted to VISAPP 2020)

Abstract

Detecting anomalies in textured surfaces is an important and interesting problem that has practical applications in industrial defect detection and infrastructure asset management with a lot of potential financial benefits. The main challenges in this task are that the definition of anomaly changes from domain to domain, even noise can differ from the normal data but should not be classified as an anomaly, lack of labelled datasets and a limited number of anomalous instances. In this research, we have explored weak supervision and network-based transfer learning for anomaly detection. We developed a technique called AnoNet, which is a novel and compact fully convolutional network architecture capable of learning to detect the actual shape of anomalies not only from weakly labelled data but also from a limited number of examples. It uses a unique filter bank initialization technique that allows faster training. For a $H \times W \times 1$ input image, it outputs a $H \times W \times 1$ segmentation mask and also generalises to similar anomaly detection tasks. AnoNet on an average across four challenging datasets achieved an impressive F1 Score and AUROC value of 0.98 and 0.94 respectively. The second approach involved the use of network-based transfer learning for anomaly detection using pre-trained CNN architectures. In this investigation, fixed feature extraction and full network fine tuning approaches were explored. Results on four challenging datasets showed that the full network fine tuning based approach gave promising results with an average F1 Score and AUROC values of 0.89 and 0.98 respectively. While we have successfully explored and developed a method each for anomaly detection with weak supervision and supervision from a limited number of samples, research potential exists in semi-supervised and unsupervised anomaly detection.

Acknowledgements

I thank my supervisor Professor John Zelek for all the guidance and support that he provided during my Master's degree.

I would also like to thank my thesis readers Professor David Clausi and Professor Oleg Michailovich for their invaluable inputs and aid.

I acknowledge and am thankful for the financial support from the Ontario Ministry of Transportation, Natural Sciences and Engineering Research Council, University of Waterloo and Systems Design Engineering department.

I thank my parents Ms. Baljeet Kaur Minhas and Mr. Amarjeet Singh Minhas for their constant love and support and sacrifices that allowed me to pursue my Master's degree. I thank my sister Harneet Kaur Minhas for her love and support.

I thank my relatives, friends, and teachers for their support.

Dedication

I dedicate this thesis to my family, friends, and teachers. First, this is for my mom, dad, and sister. Their unconditional love and support have made this a reality. Next, to all my family members and friends who are a part of my life and have supported me throughout. Finally, this is also to all the teachers I have encountered in my life who not only imparted knowledge but also life skills and wisdom to me, that allowed this day to become a reality.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

According to the World Health Organization (WHO) Global Status Report on Road Safety 2018, there are 1.35 million road traffic deaths every year [53]. A study conducted by the Pacific Institute for Research and Evaluation (PIRE) on traffic accidents and fatalities in 2009, found that more than half of the deaths that occurred on the American roadways were due to poor road conditions [46]. The expense of those accidents costs the U.S. economy $217 billion each year. Additionally, $91 billion was invested annually in road infrastructure [50]. Poor road conditions are primarily caused due to surface defects such as cracks and potholes. In the railways, broken rails and welds were the most common cause of train derailments which accounted for more than 15% of defects [51]. Detecting defects in industrial manufacturing processes is crucial for ensuring the high quality of finished products. All these examples show the vital importance of detecting defects across different industries.

A common property of these surface defects is that their visual texture is inherently different from the defect-free surface. Since human visual inspection relies solely on what is seen, it only makes sense that automating visual inspection from camera images should be plausible. The task of automated visual defect detection can therefore be formulated as the problem of anomaly detection in textured surfaces. Visual texture refers to the human visual cognition and semantic meaning of textures based on the local spatial distributions of simple appearance properties such as color, structure, reflectance, and orientation of the object. The objective of the manual human inspection is to detect the anomalies by comparing the difference in visual texture appearance of the defects against defect free appearance. The process is not only time consuming and expensive but also prone to errors due to the monotony of the task. It is also subjective and susceptible to human biases. Individual factors such as age, visual acuity, gender, experience, scanning strategy,

training, etc. also affect inspection [62]. To overcome these problems and challenges, a significant amount of work has been conducted to automate the process of anomaly detection in textured surfaces. Examples of automatic visual inspection systems in various domains include defect detection in steel surfaces [67], pavements [1], rail tracks [77] and fabric [32] to name a few. The key challenges faced by automated detection systems are as follows. Anomalies such as dents, smudges, cracks, impurities, scratches, stains, etc. vary in terms of pixel intensities, geometric constraints, and visual appearance [58]. Moreover, the data often contains noise, which although is different from the normal data, should not be classified as an anomaly by the detection system. Environmental factors such as lighting, temperature, extreme weather (such as snow) also impact the detection systems. These challenges make the task of anomaly detection in textured surfaces extremely complex and difficult.

For these automated systems, textures can be described using two approaches, namely structural and statistical. The structural approach considers texture as an organised area phenomenon which can be decomposed into primitives (also known as textons) having specific spatial distributions [20]. This definition comes directly from the human visual experience of textures. The number and type of primitives, as well as their spatial organization or layout, describes an image texture. For example, a brick wall texture is generated by tiling up bricks (primitives or textons) in a layered manner (specific spatial distribution). The second approach known as the statistical approach considers textures to be generated by a stochastic process such as a Markov Random Field. Grass, sand, sandpaper, leather, etc. are examples of this category. The quantitative measure of the spatial distribution of gray levels in textured surfaces forms the basis of the statistical approach.

Traditionally, the automated methods have relied on the computation of a set of hand-crafted textural features in the spatial or spectral domain followed by the search for significant deviations in the feature values by using different classifiers. In spatial-domain approaches, the commonly used features are second-order statistics derived from spatial gray-level co-occurrence matrices [71]. Spectral approaches normally involve the use of Gabor filters [33], Fourier transform [9] and Wavelet transform [63]. These methods, however, suffer from the following drawbacks. The hand-crafted features require domain expertise and are very challenging to formulate. They do not generalize which means that the engineered features that are designed for a specific task cannot be used for other different or even similar tasks.

Deep learning techniques applied to the task of anomaly detection have overcome these challenges and are receiving increased attention. Convolutional Neural Networks (CNNs) were used for supervised steel defect classification [42] and rail surface defect classification [11]. Although the deep learning techniques have outperformed the traditional hand-crafted

features based approaches, they suffer from their associated set of challenges. The lack of available labelled training examples is a major challenge [6] [7], since these models require large labelled datasets. The instances of anomalous classes are even fewer in these datasets which hinders the training of the network. Pixel-level annotated datasets that are required for supervised defect segmentation, are not only rarer but also expensive and time-consuming.

This research aims to develop a general anomaly detection method that can be applied to the automation of the tedious defect detection task by human inspection across different industries. Our aim is also to address the existing research gap by being able to train from a limited number of samples for the classification task and weakly labelled data for the pixel-level segmentation task. Weakly supervised learning covers techniques that try to construct models by learning with weak supervision and directly addresses these pain points. Weak supervision can be broadly classified into three categories: incomplete, inexact and inaccurate [82]. In incomplete supervision only a small subset of training set has labels and the rest of the samples are unlabelled. Inexact supervision involves coarse-grained labels, for example, image-level labels rather than object-level labels. Inaccurate supervision involves labels that are not always correctly labelled. One technique that is in both the inexact and inaccurate category is weakly supervised anomaly detection. It uses masks that are loosely annotated at the pixel level e.g. in the form of some geometric shape such as an ellipse covering the entire anomaly. As a result, the mask only provides a coarse location of the anomaly and there are a lot of inaccurately labelled normal (defect-free) background pixels which are seen as anomalous pixels. This makes the anomaly detection task even more challenging.

In this thesis, we explored two approaches. The first involved the use of weak supervision for the segmentation of anomalies using CNNs (Chapter 3) and the second involved the use of network-based transfer learning using CNNs (Chapter 4) for the classification of anomalous images from a limited number of training samples. We present a novel technique for anomaly detection in textured surfaces using weakly annotated data, that can learn the underlying shape of the anomaly from not only weakly annotated data but also from a few examples.

1.1 Contributions

The main contributions of this thesis include the following:

1. We have developed AnoNet (Chapter 3), a fully convolutional architecture with only

64 thousand parameters, for anomaly detection in textured surfaces using weakly labelled data that outputs a $H \times W \times 1$ segmentation mask for a $H \times W \times 1$ input image. This prevents the loss of localization of the anomaly with respect to the original image. The network has a valuable and important ability to learn to detect the actual shape of the anomaly despite the weak annotations.

2. AnoNet has an important practical advantage for real-world applications, that it can learn from a limited number of weakly annotated images. For the RSDDs-I dataset, it learnt do detect anomalies after just a single pass through mere 53 training images.

3. A filter bank based initialization technique for AnoNet is presented. To the best of our knowledge, no such work has been done for weakly supervised anomaly detection in textured surfaces. AnoNet achieved state of the art performance on four challenging datasets. In comparison to the CompactCNN [58] and DeepLabv3 [8], AnoNet on average achieved an impressive average improvement in performance to an F1 score of 0.98 (106% increment) and to an AUROC value of 0.94 (13% increment).

4. We have explored network-based transfer learning for anomaly detection using CNNs (Chapter 4). Results obtained on four difficult datasets showed that the full network fine-tuning based approach gave promising results with an average F1 Score and AUROC values of 0.89 and 0.98 respectively.

1.2 Thesis Outline

The remainder of the thesis is organized as follows. In Chapter 2, we discuss existing methods with their gaps and shortcomings. The AnoNet architecture (3.1) along with the overall methodology (3.2), results and discussion (3.3) are detailed in Chapter 3. Next, we describe the network-based transfer learning approach in Chapter 4. Transfer learning is discussed in 2.5, followed by the methodology (4.1), experiments (4.2) and results 4.3. Finally, in Chapter 5 we present our conclusions and future work recommendations.

Chapter 2

Background

The primary goal of defect detection and assessment is to differentiate possible defective regions from non-defective regions. For visual appearance inspection, this is just the task of anomaly detection in textured surfaces. Different and complex textures, varying shapes, sizes and colors of defects, as well as inconsistent lighting conditions, make the task extremely challenging. Traditional methods follow the pipeline of feature extraction followed by a learning based classifier such as SVM, KNN etc. for performing the task of anomaly detection. The two main drawbacks of using hand-crafted feature based approaches are the requirement of domain knowledge and poor generalization capability of the features. Deep learning techniques can overcome these challenges while achieving superior performance and have been widely investigated.

2.1 Traditional Methods vs Deep Learning

An increasing number of studies, for instance [23] [41] [57], have been carried out that compare the deep learning approaches to the traditional hand-crafted feature based techniques for different computer vision tasks. Almost all of them have one common observation that deep learning based approaches tend to work better.

In one study, Antipov et al. [3] compared learned features and hand-crafted features for the task of Pedestrian Gender Recognition. The key findings of the research were as follows. The learned features significantly outperformed the hand-crafted features on heterogeneous data composed from different datasets, with an average mean average precision (mAP) increase of 28.4% and a maximum increase of as high as 46.5%. Furthermore, the learned

features generalized better than the hand-crafted features on completely unseen datasets. An average performance improvement of 31.8% in the mAP value was observed. Finally, they found that smaller CNN models trained from scratch on small datasets were able to produce compact features that generalized as well as the features produced by much bigger pre-trained networks fine-tuned on the same datasets.

In their work, Nanni et al. [47] compared handcrafted and non-handcrafted features for classification on several datasets. They extracted features from different intermediate layers of CNN models in comparison to traditional descriptors like Local Binary Patterns, Local Ternary Patterns, and Local Phase Quantization by training SVMs for the classification task. Results showed that the deep learning feature based approach performed better.

In [5], the authors compared learned and handcrafted feature based approaches for person re-identification. They found that fully trained CNN outperformed the handcrafted approaches and the combination of pre-trained CNN with different re-identification processes. However, they identified that the deep learning methods tended to over-fit on single-shot databases (which is to be expected since it comprises of only one image pair) and required large training samples, high computational power as well as longer training time.

Zare et al. [78] explored and compared three different approaches for the classification of Medical X-ray images, namely (1) SVM trained on features extracted from bag-of-visual-words (BoVW) model (2) SVM trained on features extracted from pre-trained AlexNet and (3) fine-tuning of pre-trained AlexNet using transfer learning. Results on the ImageCLEF 2007 dataset with 116 classes showed that the fine-tuning approach outperformed the other techniques by achieving per class accuracy of greater than 80% in 60 classes compared to just 24 and 26 classes for first and second technique respectively. Because of the clear performance improvement, recently, deep learning based approaches for anomaly detection have gained a lot of attention.

2.2 Anomaly Detection Techniques

The anomaly detection techniques can be broadly classified into three major categories based on the availability of labels: (1) unsupervised (2) semi-supervised and (3) supervised [7] [6]. The class labels can be either normal or anomalous.

1. **Unsupervised anomaly detection**: These methods do not require any training labels which makes them the most flexible. They make use of just the intrinsic properties of the data with the assumption that the data is heavily skewed with a lot more normal instances than the anomalous. In [56], the authors explored an image segmentation approach for fruit defect detection using k-means clustering and graph-based algorithm. The method employed a region growing technique for the segmentation process making it slow. Also, it was heavily dependent on the choice of the initial cluster centers and got different results based on different selections. With unsupervised approaches, one cannot hope to target specific types of anomalies since there is no labelling. Also, although the unsupervised techniques offer the most flexibility in terms of labelling requirements, they often struggle to learn commonalities within complex and high dimensional spaces [6].

2. **Semi-supervised anomaly detection**: The main assumption for these techniques is that we have access to labelled training instances for only the normal class. Because of this reason, these techniques are less flexible than unsupervised techniques. These try to estimate or model the underlying normal class distribution of the data. This is followed by some kind of measurement of divergence or deviation of the samples from the learnt distribution and classification based on a threshold. Schlegl et al. [59] used Generative Adversarial Networks (GANs) for anomaly detection in optical coherence tomography images of the retina. They trained a GAN on the normal data to learn the underlying distribution to encode the anatomical variability. However, they did not train an encoder for mapping the input image to the latent space. As a result, the method needed an optimization step for every query image to find a point in the latent space that corresponded to the most visually similar generated image. This caused the technique to be computationally expensive. The model outputted an anomaly score and a segmentation output as the residual image obtained by subtracting the query image from the GAN generated image from optimization. Zenati et al. [80] tried to handle the speed bottleneck by training an encoder that learnt the mapping from the data space to the latent space. They tested it on only two image datasets SVHN and CIFAR10 and the results did not look good. They got AUROC values of 0.5753 and 0.6072 respectively which are marginally above the random classification model value of 0.5. In [76], the authors used convolutional auto-encoders for defect segmentation of hot-rolled steel strip surfaces. They trained an auto-encoder on only the normal images. The anomaly detection process involved a sharpening step which was a weighted addition of the residual image (obtained by subtracting the reconstructed image from the input image) to the reconstructed image. This was followed by Gaussian blurring and thresholding. No quantitative

results were reported. However, the qualitative results showed that the technique was extremely noisy. It segmented even illumination changes as defective regions which were undesirable and ultimately affected the performance.

3. **Supervised anomaly detection**: These techniques require labels of both normal and anomalous data instances which makes them the least flexible. However, the supervised techniques tend to have better performance in comparison to the other two types. Since the training is done on labelled instances, certain specific types of defects or anomalies can be targeted. This could be especially helpful in industrial defect detection and infrastructure asset management, where certain types of defects may be extremely detrimental to the working or safety of the product or equipment or asset and need to be identified with high precision. As a result of these benefits, a growing body of literature has examined these techniques. Specifically, the use of Convolutional Neural Networks (CNNs) for supervised anomaly detection has seen an exponential increase, which is discussed next.

2.3 Anomaly Detection using CNNs

One of the primary reasons behind the increased adoption of CNNs is their ability to eliminate the need for domain-specific engineered features by learning complex filters from the training data. In [42], CNNs were used with max pooling layers for the task of steel defect classification into 7 defect types and their performance was compared with SVM classifiers trained on engineered features. The CNN based approach performed approximately two times better in comparison to the feature descriptor based SVM approach. The best CNN model having 7 hidden layers had an error of 6.79%, while the best feature i.e. Pyramid of Histograms of Orientation Gradients (PHOG) based classifier had an error of 15.48%. Another important observation was that the network with large filter sizes did not achieve the best performance. The network with progressively decreasing filter sizes of $11 \times 11, 6 \times 6$ and 5×5 achieved better performance than the network with 19×19 and 13×13 filter sizes. Their technique, however, had the following three shortcomings. First, the input image size to the network was restricted to one value because of the use of fully connected layers. Next, the network did the classification without localizing the defect in the images via a pixel-level segmentation mask. Finally, since a large number of kernels per CNN layer were used, this increased the number of network parameters and potentially exposed the models to the problem of over-fitting. Over-fitting is a problem in deep learning where the model is more complicated than is necessary, thereby leading to the memorization of datasets. Even though an over-fit model can have a good performance on the training

data, it does not generalize well and has poor performance on the testing data and does not scale to other testing data.

Another major challenge for the CNN based approaches applied to supervised anomaly detection is that they require a large number of samples of normal and anomalous instances for training [7] [6]. Specifically, detailed pixel level annotation of the anomalous instances is required for the segmentation tasks. For all practical applications and purposes, this is a major drawback. This is because not only are anomalous instances limited in real-world applications, but also the creation of pixel-level annotated datasets is cumbersome and expensive. A technique to tackle this challenge was explored on different CNN architectures for the task of surface anomaly detection [73]. To train the network, the proposed method required sub-sampling of the original images with the extraction of 32×32 patches which lead to a 47 fold increment in the number of training samples. However, this sub-sampling approach resulted in extremely long training time of 24 hours and also led to the loss of the global contextual information required for the anomaly detection task. Also, patch based approaches are several times slower than the FCN based approaches where the entire input image is processed in a single forward pass through the network.

2.4 Weakly Supervised Anomaly Detection

One way of tackling the challenge of lack of labelled data is to use weakly supervised learning methods. As discussed in the previous chapter, these techniques try to construct models by learning from weak supervision. However, very few techniques in the literature exist that tackle anomaly detection using weak supervision. In [58], the authors used a CNN based architecture for the classification and segmentation of anomalies from weakly annotated data. However, their approach had the following shortcomings. The network did not learn to detect the actual shape of the anomaly from the weak labelling. The input size of the image for the model was fixed to 512×512, which prevented images of other shapes to be fed to the network. It outputted masks of size 128×128, which were sixteen times smaller than input image size and resulted in the loss of localization and shape information. This can introduce errors in the calculation of metrics which are based on the shape and size of the defect (anomaly). Also, the network was not tested on any real world dataset which raised concerns regarding its practical application. The paper lacked quantitative analysis for the segmentation task. The qualitative results presented and discussed in the paper showed that the segmentation results were poor, with a lot of false positive pixels i.e., defect free regions classified as defects. The classification part of the model completely relied on the features extracted by the model for the segmentation

task. Thus, good segmentation capability was essential for the classification stage. Lastly, the proposed architecture had approximately 1.13 million parameters. The huge number of parameters made the architecture susceptible to the problem of over-fitting. If the optimization problem is made easier by changing model architectures (by making them deeper and thereby increasing the total number of model parameters), generalization performance can be degraded [27].

2.5 Anomaly Detection using Transfer Learning

Transfer learning is another technique that is used to handle situations where you do not have access to large labelled datasets. The goal of transfer learning is to improve learning in a target task by leveraging knowledge from a source task [70]. Chuanqi Tan et al. [69] classify deep transfer learning into four categories: (1) instance-based deep transfer learning, (2) mapping-based deep transfer learning, (3) network-based deep transfer learning, and (4) adversarial based deep transfer learning. Out of these approaches network-based deep transfer learning method is most widely used in practical applications. It refers to the reuse of a partial network pre-trained for a source domain, including its network structure and connection parameters and transferring it to be a part of deep neural network which used for a target domain [69]. The source network is thought of as consisting of two sub-networks: (1) Feature extractor sub-network and (2) Classification sub-network. The target network is constructed using the source network with some modifications and trained on the target dataset for the intended task. The network-based transfer learning approach is shown in Figure 2.1.

A growing body of literature has examined the use of transfer learning for different classification tasks. Kensert et al. applied transfer learning for classifying cellular morphological changes and explored different CNN architectures [28]. The ResNet50 architecture achieving the highest accuracy of 97.1%. They observed that the models were able to distinguish the different cell phenotypes despite a limited quantity of labelled data. In another study, Feng et al. [14] used transfer learning for structural damage detection. The Inception-v3 architecture obtained an average accuracy of 96.8% using transfer learning and outperformed the SVM method which had an accuracy of 61.2%.

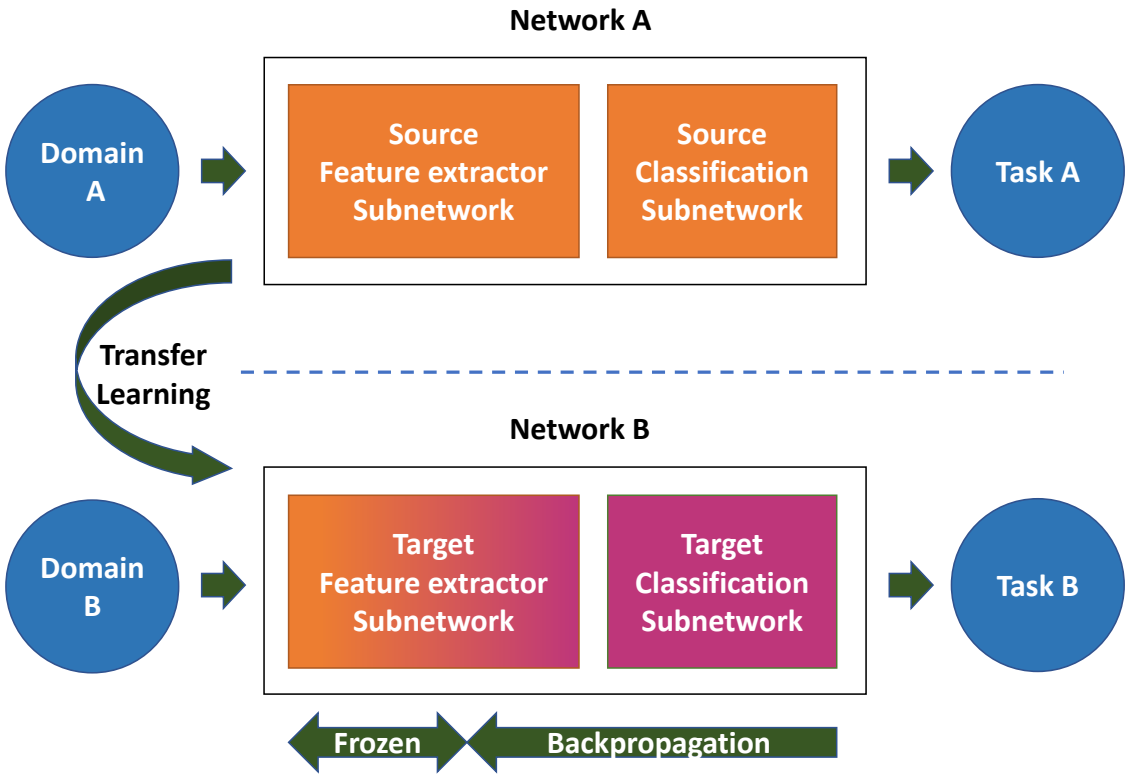

Figure 2.1: Illustration of a network based deep transfer learning from a source domain A and task A to target domain B and task B. The Network A is trained on a large training dataset and is called the pre-trained network. Network B is constructed by using parts of Network A followed by a new softmax classification network. Finally, the resulting network B is initialized with the pre-trained weights and trained using backpropagation on the target dataset.

2.6 Summary

To summarize, anomaly detection is a very relevant problem with applications across industries and specifically in defect detection and infrastructure asset management and maintenance. However, it has a lot of associated challenged. Different lighting conditions, complex textures, and varying shapes, sizes and colors of the defects are some of them. Limited instances of anomalies and lack of labelled data add to the difficulty of learning

algorithms.

Recently, with the proliferation of data, access to powerful processors, better network architectures and improved optimization techniques, deep learning techniques have gained a lot of success. For computer vision tasks, Convolutional Neural Networks have outperformed the traditional hand-crafted feature based approaches. But the requirement of large amounts of training data remains a huge challenge. The situation is aggravated for anomaly detection because of the lack of labelled data and the unavailability of a large number of anomalous examples.

One type of technique to handle this challenge is to use weak supervision. For anomaly detection, this is in the form of imprecise and inexact training data e.g. an ellipse covering the entire defect such as a crack. A network architecture that learns to detect the actual shapes of the anomalies from limited samples of weakly labelled data, does not over-fit and generalizes to similar anomaly detection tasks is missing in the literature. In this research, we present a novel technique (Chapter 3) capable of learning to detect the actual shape of the anomalies from not only weakly labeled data but also from a limited number of samples. To achieve this, we explore pre-seeding the preliminary feature extractor with a biologically plausible one. Empirical testing of the architecture is used to find a compact design.

Transfer learning is another technique that also tries to address the challenge of training from limited data. Although transfer learning for classification has been explored for specific applications, an extensive exploration of anomaly detection using transfer learning comparing the performance of the state-of-the-art CNN architectures on different defect detection tasks is missing in the literature. In this research, we uniquely use the output value from the neuron responsible for the anomalous samples as the anomaly score value. And the approach (Chapter 4) was tested on three different CNN architectures and four challenging datasets. Unlike the current work on defect detection using transfer learning, we use the AUROC (3.2.7) metric for evaluating the model performance, because it is a robust and more accurate measure of the separation capability than just the classification accuracy.

Chapter 3

AnoNet: Weakly Supervised Anomaly Detection in Textured Surfaces

In this chapter, first the AnoNet architecture is explained along with the filter bank initialization technique. This is followed by the methodology and results.

3.1 AnoNet: A fully convolutional network for anomaly detection

3.1.1 Network Architecture

The AnoNet architecture is presented in Figure 3.1, it is a modification of the CompactCNN [58]. It is a Fully Convolutional Network (FCN) and therefore overcomes the restriction of a fixed size input faced in the case of CNNs that make use of fully connected layers. AnoNet consists of four convolutional layers and all the layers have a stride of one. For all the layers a zero padding of $\frac{k-1}{2}$ (where $k \times k$ is the kernel size of the layer) is done on all the sides to ensure that the size of the output feature maps is same as the input feature maps. Progressively decreasing filter sizes of $k \times k$ ($k \in \{11, 7\}$), 7×7, and 3×3 are used to allow the network to have a large field of view, which is beneficial for the anomaly detection task. The first layer of AnoNet is the filter layer which is seeded using the Filter Bank initialization technique 3.1.2, which is biologically plausible. The rest of the layers

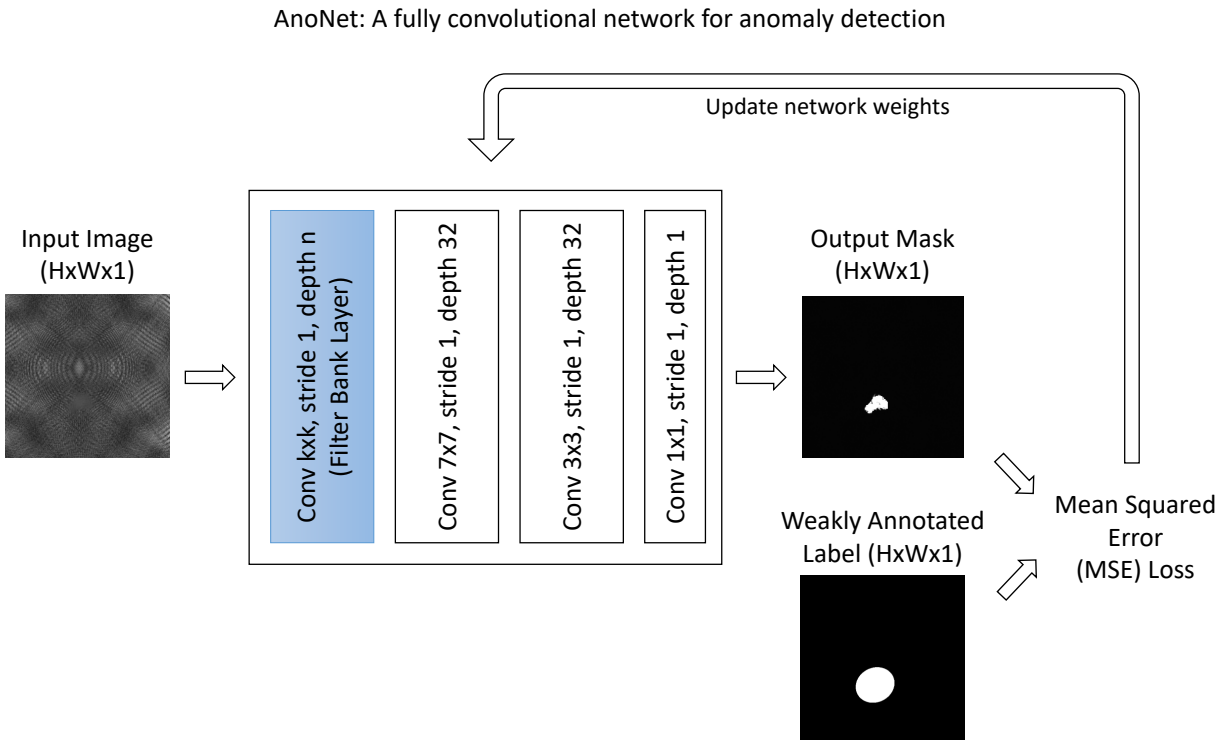

Figure 3.1: AnoNet: A fully convolutional network, for anomaly detection in textured surfaces using weakly labelled data that outputs a $H \times W$ segmentation mask for a $H \times W$ input image. With only 64 thousand parameters, AnoNet is remarkably compact and not susceptible to the problem of over-fitting. It has the valuable and important ability to learn to detect the actual shape of the anomaly not only from the weak annotations but also from a limited number of samples. It uses a filter bank initialization technique for the first layer, as described in 3.1.2 and the values of network parameters k(filter size) and n (filter stack length) depend on the filter bank being used and are described in subsection 3.1.2. For the rest of the network the weights are initialized using a random normal distribution of mean zero and variance of one.

are initialized with a random normal distribution with zero mean and variance of one. The values of the parameters k (filter size) and n (filter stack length) depend on the AnoNet configuration and are summarized in Table 3.1. All the layers except the last layer use ReLU activation function which is defined by the Equation 3.1. It can be shown that deep neural networks trained with ReLU train several times faster than their equivalents with

tanh units [31].

$$f(x) = max(0, x) \tag{3.1}$$

However, for the segmentation layer (i.e. the last layer), the tanh activation is used. It was selected since it resulted in a better separation of the anomalies from the normal pixels in comparison to the ReLU and linear activation. Batch normalization is applied after every layer since it accelerates the training of deep networks by making normalization inherent to the model architecture [26]. Normalization of a vector means making it have the mean of zero and a variance of one. For a vector $\mathbf{x}$, the normalization equation is given below by Equation 3.2, where $E[\mathbf{x}]$ is the expectation of $\mathbf{x}$ and $Var[\mathbf{x}]$ is its variance.

$$\mathbf{x} = \frac{(\mathbf{x} - E[\mathbf{x}])}{\sqrt{Var[\mathbf{x}]}} \tag{3.2}$$

In comparison to the segmentation part of CompactCNN [58], the AnoNet architecture achieves a massive reduction of 94.29% in the total number of network parameters from 1.13 million to 64 thousand on average. Despite this huge reduction in parameters, AnoNet outperformed CompactCNN in the anomaly detection task as shown in Section 3.3. For a $H \times W$ image, the network outputs a $H \times W$ mask. This is because down-sampling operations such as strided convolutions and pooling have not been performed in the network architecture. This also prevents the artifacts that are introduced during the up-sampling transposed convolution or deconvolution operation. AnoNet has the valuable ability to learn to detect the actual shape of the anomalies from weakly annotated datasets with a limited number of training samples.

The unique features of the AnoNet architecture are as follows.

1. AnoNet is a fully convolutional network and does not use strided convolution (i.e., layers with stride > 1) which does not down-sample the image. For a $W \times H$ input image, we get a $W \times H$ output mask. Since the model does not use transposed convolutions for up-sampling, there are no checkerboard artifacts.

2. The network is shallow and compact which prevents over-fitting by design. Additionally, this allows the training of the network accomplished with only a limited number of training samples.

3. The compactness of the model causes the size of the intermediate features to be limited which allows the training to be done without having to down-size the image to a lower resolution before making the batches to feed to the GPU.

Table 3.1: AnoNet: Filter Bank configurations. There are 12 AnoNet configurations in total and their names are given in the configuration column. The filter bank column refers to the filter bank being used. Filter size column gives the AnoNet parameters k and n. The trainable column contains Boolean values. True means that filter layer (first layer) of AnoNet was set to be trainable i.e., its parameters were updated during the training and False means that the parameters were frozen and did not change during the training.

Configuration	**Filter Bank**	**Filter Size** $(k \times k \times n)$	**Trainable**
LMExp1	LM	7x7x48	False
LMExp2	LM	7x7x48	True
LMExp3	LM	11x11x48	False
LMExp4	LM	11x11x48	True
RFSExp1	RFS	7x7x38	False
RFSExp2	RFS	7x7x38	True
RFSExp3	RFS	11x11x38	False
RFSExp4	RFS	11x11x38	True
SExp1	S	7x7x13	False
SExp2	S	7x7x13	True
SExp3	S	11x11x13	False
SExp4	S	11x11x13	True

4. The model footprint is small which makes it suitable for local execution on edge and IoT devices.

5. The network can learn to detect the underlying shape of the anomaly despite the weak labelling.

3.1.2 Filter Bank Initialization Technique

Filter banks usually refer to a collection of specially designed hand-crafted kernels that are stacked together and applied to images to extract useful features for a particular task. This is usually followed by the use of a learned classifier such as an SVM to perform the classification or segmentation. Gabor filters [33], Wavelet filters [63] and Difference of Gaussians are examples of some commonly used filters for texture related tasks. In our proposed technique, three specific filter bank sets namely the Leung-Malik (LM), Schmid

(S) and Root Filter Set (RFS) were selected since they contained both rotationally invariant as well as directional filters [72] [34] [60] [17]. Thus, they are general and some have claimed that they are biologically plausible [61]. Images of each of these three filter banks extracted at an 11x11 kernel size are shown in Figure 3.2.

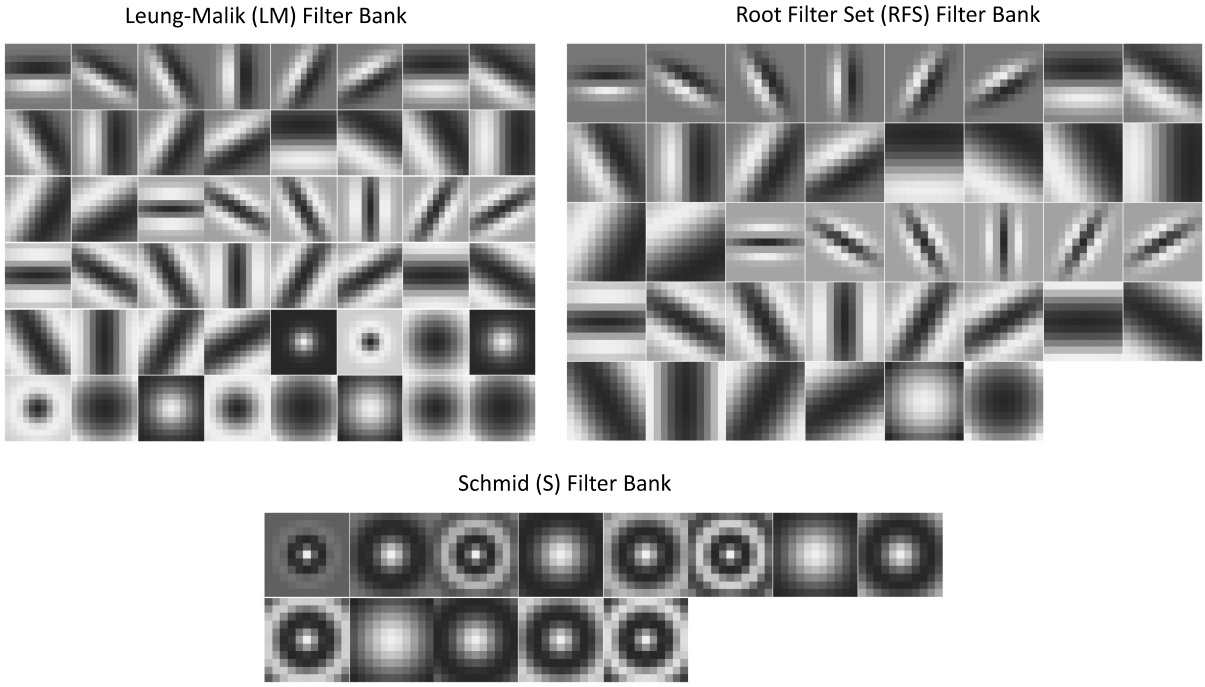

Figure 3.2: The figure shows the LM, S and RFS filter banks extracted at an 11x11 kernel size. The LM filter bank has a mix of edge, bar and spot filters at multiple scales and orientations. It consists of first and second derivatives of Gaussians at 6 orientations and 3 scales making a total of 36; 8 Laplacian of Gaussian (LOG) filters; and 4 Gaussians. The RFS filter bank consists of 2 anisotropic filters (an edge and a bar filter, at 6 orientations and 3 scales), and 2 rotationally symmetric ones (a Gaussian and a Laplacian of Gaussian). The S filter bank consists of 13 isotropic Gabor like filters [40].

1. **Leung-Malik (LM) Filter Bank**: The LM filter bank comprises of a set of 48 multi-scale and multi-orientation filters. There are 36 filters of 1^{st} and 2^{nd} order derivatives of Gaussians at 6 orientations and 3 scales, along with 8 Laplacian of Gaussian (LOG) filters and 4 Gaussians filters [40].

2. **Schmid (S) Filter Bank**: The S filter bank comprises of 13 rotationally invariant filters which have the following form shown in Equation 3.3 [40].

$$F(r, \sigma, \tau) = F_0(\sigma, \tau) + cos(\frac{\pi \tau r}{\sigma})\ e^{-\frac{r^2}{2\sigma^2}} \tag{3.3}$$

The $F_0(\sigma, \tau)$ term is added to make the DC component zero. The rotational symmetry of this filter bank can be seen in Figure 3.2.

3. **Root Filter Set(RFS) Filter Bank**: As shown in Figure 3.2, the RFS filter bank is similar to the LM filter bank. It comprises of 38 filters and uses a Gaussian and a Laplacian of Gaussian both with $\sigma = 10$ pixels, an edge filter at three scales $(\sigma_x, \sigma_y) = \{(1, 3), (2, 6), (4, 12)\}$ and a bar filter at the same three scales [40].

3.2 Methodology

Our investigation was conducted in four stages: (1) analysis of CompactCNN, (2) visualization studies, (3) ablation studies, and (4) AnoNet filter bank studies.

3.2.1 Datasets

Four datasets were selected for experimentation. The dataset selection contained one artificially generated dataset and three real world datasets all with completely different textures and defects. Each dataset had a limited number of training samples which made the anomaly detection task difficult. Additionally, varying illumination, different camera positions, and orientation added to the complexity. The wide variety and challenging nature of these datasets ensured that the proposed technique was tested thoroughly and not limited to any particular type of texture and defect.

The datasets we used include the following.

1. **DAGM**[44] is a synthetic dataset for weakly supervised learning for industrial optical inspection. It contains ten classes of artificially generated textures with anomalies. For this study, the Class 1 having the smudge defect was selected, since it had the maximum intra-class variance of the background texture of all classes in the dataset. It (hereafter referred to as DAGMC1) contains 150 images with one defect per image and 1000 defect free images. For every image a weakly labelled annotation in the form

of an ellipse that covers the entire defect is available. The ellipse covers a significant amount of normal texture in addition to the defect making the dataset an excellent test case for loosely labelled data.

2. **CrackForest** [64] dataset consists of urban road surface images with cracks as defects. The images contain confounding regions such as shadows, oil spills, and water stains. The images were taken using an ordinary iPhone5 camera. The dataset contains 118 images and has corresponding pixel level masks for the cracks, all having a size of 320×480. The additional confounders along with the limited number of samples available for training make CrackForest another good dataset for anomaly detection evaluation.

3. **Magnetic Tile Defects dataset** [25] dataset contains images of magnetic tiles collected under varying lighting conditions. Magnetic tiles are used in engines for providing constant magnetic potential. There are five different defect types available, namely Blowhole, Crack, Fray, Break and Uneven. Among these, blowholes and cracks impact the quality of magnetic tiles the most. We use the Blowhole category (referred to as MT_Blowhole) of this dataset since CrackForest already covers a crack type defect. The Blowhole defect category contains 115 images of varying sizes and pixel level annotations are available for the defects.

4. **RSDDs (Rail surface discrete defects)** [15] is a challenging dataset containing varying sized images of two different types of rails. Rail surface defects are one of the most common and most important forms of failure [15]. Every image contains at least one defect and has a complex background with noise. The RSDDs Type-I category contains 67 images from express rails and the Type-II category contains 128 images captured from common/heavy haul rails. Pixel level annotations are available for the defects for both categories. The heavily skewed aspect ratio of the images and a limited number of training samples make this dataset challenging for the task of anomaly detection.

To ensure that the datasets had weakly labelled annotations, all the datasets except DAGMC1 (since it was already weakly labelled) were modified by performing the dilation operation using an 11×11 filter. A sample image and weakly annotated mask pair from each dataset are shown in Figure 3.3.

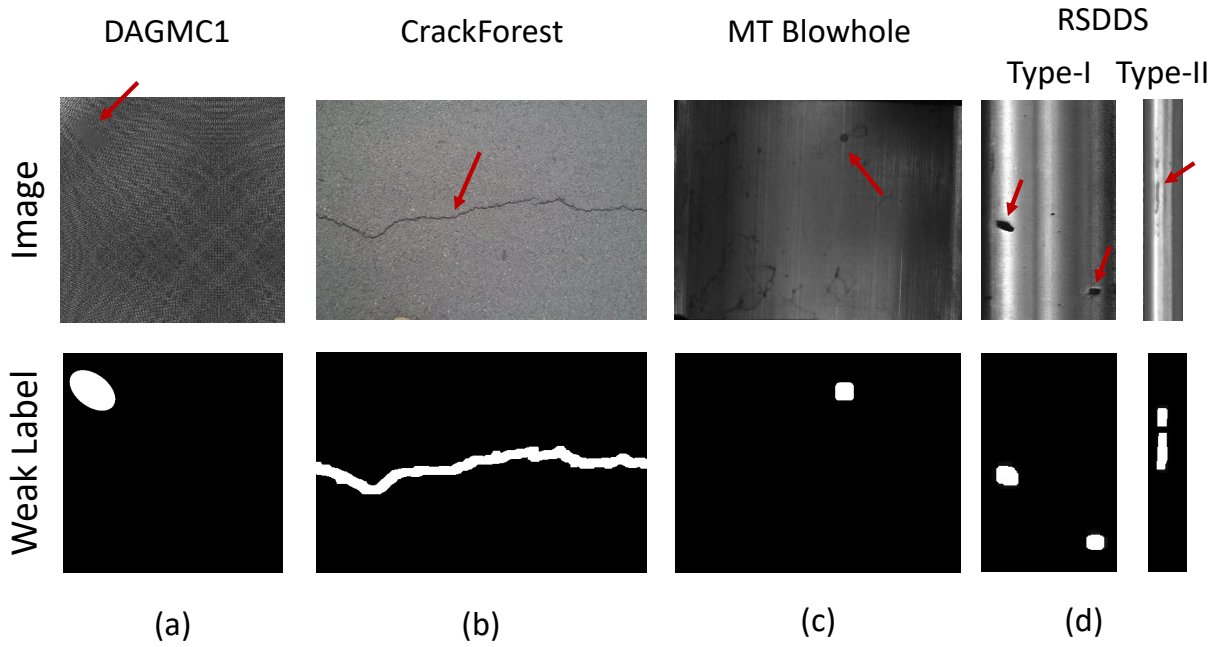

Figure 3.3: Figure shows one sample image and weakly labelled mask pair per dataset. The red arrows point toward the anomaly in the images. Figure 3.3 (a) shows the sample from DAGMC1 having a smudge as the anomaly. Figure 3.3 (b) shows a sample from CrackForest dataset and has cracks as the anomaly. Figure 3.3 (c) shows the MT Blowhole sample. The defect or anomaly is a Blowhole, a type of surface defect in magnetic tiles. Figure 3.3 (d) shows the RSDDS-I and RSDDS-II dataset samples. These have surface defects on express rails and common/heavy haul rails respectively as the anomaly. Dilation was performed for all the datasets except DAGMC1 by using an 11×11 filter to make the masks weakly annotated. The resultant weakly annotated mask examples are shown in this figure.

3.2.2 First Stage: Analysis of CompactCNN

CompactCNN [58], a CNN based architecture was presented for the segmentation and classification of anomalies in textured surfaces from weakly annotated data. As discussed in Section 2, it failed to learn the actual shape of the anomaly from the weak annotation and could not learn from a limited number of training samples. The following modifications were performed to the segmentation part of this network to overcome its limitation of fixed size input and to investigate a different activation function for the segmentation layer.

1. To overcome the restriction of a fixed size input to the network, the fully convolutional segmentation part of the network presented in [58] was selected and its input was kept as $H \times W \times 1$. (In TensorFlow, None x None x 1 was used in the placeholder to infer the dimension of the tensor from the data.)

2. The tanh activation was chosen for the segmentation layer instead of the linear activation used in [58], because a linear activation gave poorer segmentation output masks in comparison with the tanh activation. The tanh activation restricted the output to $[-1, 1]$, thereby enabling the network to better separate the anomalies from the normal texture.

Initial investigative experiments were conducted using the modified CompactCNN architecture (Figure 3.4) on the DAGMC1 and CrackForest dataset and the results are discussed in Section 3.3.

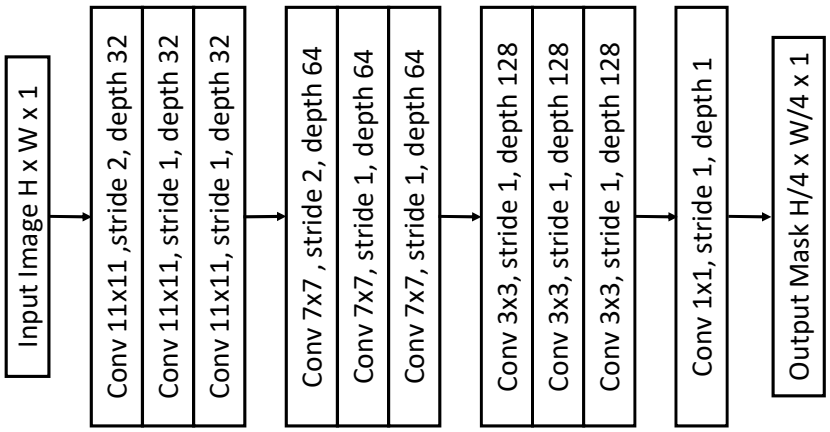

Figure 3.4: The base architecture which is a modified version of the segmentation part of architecture presented by Raĉki et al. [58]. It was pruned through extensive ablation studies to get the AnoNet architecture.

3.2.3 Second Stage: Visualization Studies

Although deep learning models tend to give superior performance, interpreting why they work is not self explanatory. To build trustworthy systems and enable their meaningful integration into the industry, model interpretability is important. The deeper the model, the more difficult it is to interpret the results. To check whether the model learnt meaningful

filters and analyse why the model failed to learn the underlying shape of the anomalies, the following visualization and activation maximization studies were conducted.

1. **Visualization of intermediate layer outputs:** Observing the intermediate layer outputs gives an understanding of the features being extracted by the kernels for a given input. This study was conducted using the following steps.

   i. Forward pass an input image X through the network.

   ii. Extract the intermediate activation $a_i^j(X)$ for i^{th} filter of layer j, for all the filters of every layer of the model.

   iii. Stack all the intermediate activation outputs and analyse them in a grid.

2. **Activation Maximization Study:** Deep learning and human brain analogies are very popular. Certain stimuli can cause specific cells in the brain to have a high response and are known as their preferred stimuli. These preferred stimuli are used in neuro-science to understand the brain. A similar activity for a deep learning model can give us a better understanding of what a neuron or kernel is doing. This technique for deep learning models is known as activation maximization [49]. Finding an image X that maximizes the activation $a_k^l(X)$ for k^{th} filter of layer l, can be formulated as following optimization problem.

$$X^* = \arg\max_X(a_k^l(X))) \tag{3.4}$$

   The following steps were used for conducting the activation maximization experiments.

   i. Randomly initialize an input image X.

   ii. Define the optimization loss as the mean value of the activation of a particular filter of a specific layer.

   iii. Calculate the gradients of the input image with respect to the loss.

   iv. Perform gradient ascent for n steps using a learning rate α on the input image to maximise the activation of the filter.

After this stage, the ablation studies were conducted.

3.2.4 Third Stage: Ablation Studies

Ablation study is a concept also borrowed from neuro-science and it refers to the selective removal or destruction of tissue to understand its function. In the context of neural networks, ablation studies in the literature are conducted by removing parts, tweaking layers and changing the structure or implementation of neural networks to assess the corresponding changes in the network performance [36]. Such ablation studies were conducted starting with the modified CompactCNN architecture. In total nine configurations were used for the experiments which are summarized in Table 3.2. Every ablation configuration had three convolutional blocks similar to the network shown in Figure 3.4. In Table 3.2, stride refers to the stride value in the first layer of the first and second blocks of the model respectively. Layers per block imply the number of convolutional layers in the block. Filter sizes are given per block from the first to the third block. Filter depth also follows the same order. For a $H \times W$ input image, for the first two experiments, the output size is $\frac{H}{4} \times \frac{W}{4}$, while for the rest of the experiments, the output size is $H \times W$. To achieve a selective reduction in the number of network parameters, the number of layers per block were gradually reduced from Exp1 to Exp5. We hypothesized that the reduction in the network parameters would address the problem of over-fitting. The distribution of the total number of network parameters for all the configurations is shown in Figure 3.5. Starting with the ablation configuration Exp2, there was an intended exponential decrease in the number of network parameters. The maximum reduction in parameters was obtained in the Exp6 configuration, with a decrease of 98.3% in comparison to CompactCNN. Different kernel size combinations were tested in the configurations from Exp6 to Exp9. From the experimental results (Section 3.3), Exp4 was found as the optimum configuration and it was selected as the AnoNet architecture. After this selection, we proceeded with the AnoNet Filter Bank studies which are described in the next subsection.

3.2.5 Fourth Stage: AnoNet Filter Bank Studies

The twelve configurations used for the filter bank studies are presented in Table 3.1. The AnoNet architecture used for these studies is discussed in Section 3.1 and shown in Figure 3.1. The parameter n in this network was set as per the filter bank stack length while the parameter k depended on the filter bank kernel size. The filter banks were extracted at 11×11 and 7×7 kernel sizes. The extracted filter bank values were used to initialize the weights of the first layer of AnoNet. In Table 3.1, the configuration column contains the names of the AnoNet configurations, the filter bank column contains the filter bank type, filter size gives the parameters k and n of AnoNet. The trainable column contains

Table 3.2: Ablation study configurations: There were 9 configurations that had three convolutional blocks each. The configuration column gives the name of the configuration. Stride refers to the stride value in the first layer of the first and second blocks of the model respectively. Layers per block imply the number of convolutional layers in the block. Filter sizes are given per block from the first block to the third block. Filter depth also follows the same order. For the first two experiments, the output size is $\frac{N}{4} \times \frac{N}{4}$ and for the rest of the experiments, the output size is $N \times N$.

Configuration	Stride	Layers per Block	Filter Sizes	Filter Depth Per Block
Exp1	2	3	11,7,3	32,64,128
Exp2	2	2	11,7,3	32,64,128
Exp3	1	1	11,7,3	32,64,128
Exp4	1	1	11,7,3	32,32,32
Exp5	1	1	11,7,3	8,32,32
Exp6	1	1	3,3,3	32,32,32
Exp7	1	1	7,7,7	32,32,32
Exp8	1	1	11,11,11	32,32,32
Exp9	1	1	3,7,11	32,32,32

Boolean values where True indicates that the filter layer (first layer) of AnoNet was set to be trainable, i.e., its parameters were updated during the training and False indicates that the parameters were frozen and did not change during the training and thus acted similar to fixed feature extractors. To compare the performance of AnoNet with the state of the art segmentation networks, DeepLabv3 [8] was selected as a representative network pre-trained for the semantic segmentation task. The segmentation head of the network was modified according to the anomaly detection segmentation task (to output a binary mask per image) and fine tuned for all of the datasets for comparative analysis.

3.2.6 Experimental setup

For the experiments, an NVIDIA Titan Xp graphics card was used. The experiments were conducted using TensorFlow version 1.12. Adadelta optimizer [79] was used with the default settings. The input size to all the network configurations was kept as $(None, None, 1)$. A batch size of 16 was used for all the experiments. All the network weights were initialized

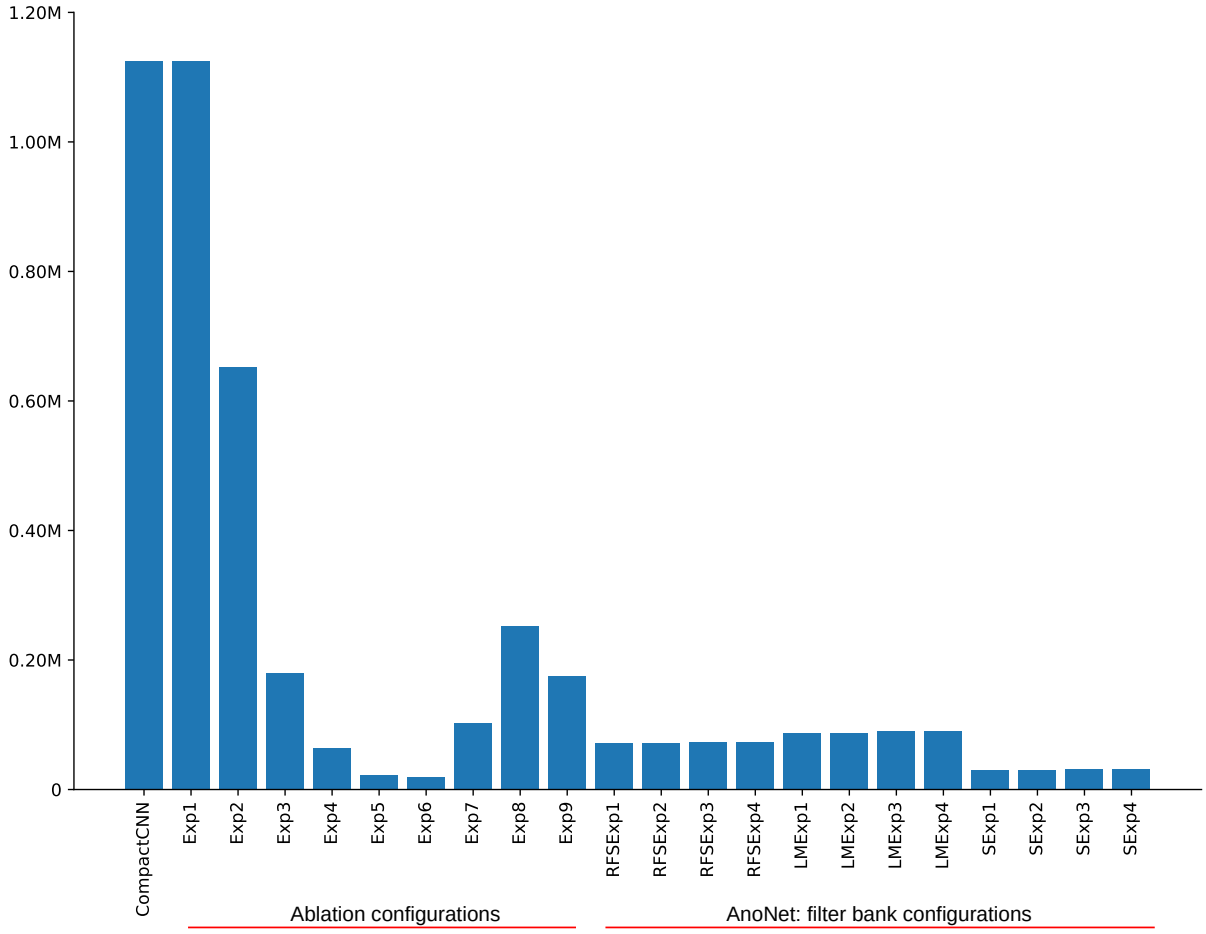

Figure 3.5: Total network parameter comparison for all the configurations used in the ablation and filter bank studies. The AnoNet architecture achieved a reduction of approximately 94% in the total number of parameters in comparison to the CompactCNN [58].

as proposed in [21]. All the experiments were conducted for 25 epochs. For the calculation of the F1 score, a threshold value of zero was used across all the experiments. The loss function used for the ablation and AnoNet filter bank experiments was the MSE (Mean Squared Error) which is defined in Equation 3.5.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{3.5}$$

As discussed in subsection 3.2.1, the dilation operation was performed on all the datasets except DAGMC1, using an 11×11 filter to make the masks weakly annotated. This ensured that normal pixels were included in the masks. The ablation and filter bank experiments were conducted as per the configurations discussed in subsection 3.2.4 and 3.2.5 respectively.

3.2.7 Evaluation Metrics

To evaluate the quantitative performance of the models, two metrics were selected. The first metric was the area under curve (AUC) measurement of the receiver operating characteristics (ROC) [37]. AUC or AUROC is a reliable measure of the degree or measure of the separability of any binary classifier (binary segmentation masks in this case). It provides an aggregate measure of the model's performance across all possible classification thresholds. An excellent model has AUROC value near to the one and it means that the classifier is virtually agnostic to the choice of a particular threshold. The second metric used for the assessment was the F1 score. It is defined as the harmonic mean of precision (P) and recall (R) and is given by the Equation 3.6. F1 score reaches its best value at one and the worst score at zero. It is a robust choice for classification tasks since it takes both the false positives and false negatives into account.

$$F1 = 2 \times \frac{P \times R}{P + R} \tag{3.6}$$

3.3 Results

This section is organised as the discussion of the results of the first to the fourth stages, namely Analysis of CompactCNN, Visualization Studies, Ablation Studies, and AnoNet Filter Bank Studies. It is important to note that for calculating the F1 score, a threshold of zero was used for all the experiments because that is the mean value of the tanh activation's range. The F1 score value can vary depending on the choice of the threshold. However, the AUROC metric takes into account all the possible thresholds into its calculation. Since the DAGMC1 dataset contains defect free examples, AUROC values cannot be calculated for this dataset.

3.3.1 First Stage: Analysis of CompactCNN

Experiments were conducted on the DAGMC1 dataset using the modified CompactCNN architecture. The results showed that the modifications led to a significant improvement in the F1 score from 0.04 to 0.97 (a threshold of zero was used). It produced better qualitative segmentation results than the ones presented in [58]. However, even for the modified architecture, the segmentation shape of the anomalous region was oval and over dilated just like the weakly labelled masks used for training and is shown in Figure 3.6. To test and check whether the architecture works on real-world datasets, experiments were conducted on the CrackForest dataset [64]. The model achieved an impressive F1 score of 0.901. Next, to further analyse why the model failed to learn the underlying shape of the anomalies and get a deeper understanding of the learnt features, visualization studies were conducted and the results are discussed next.

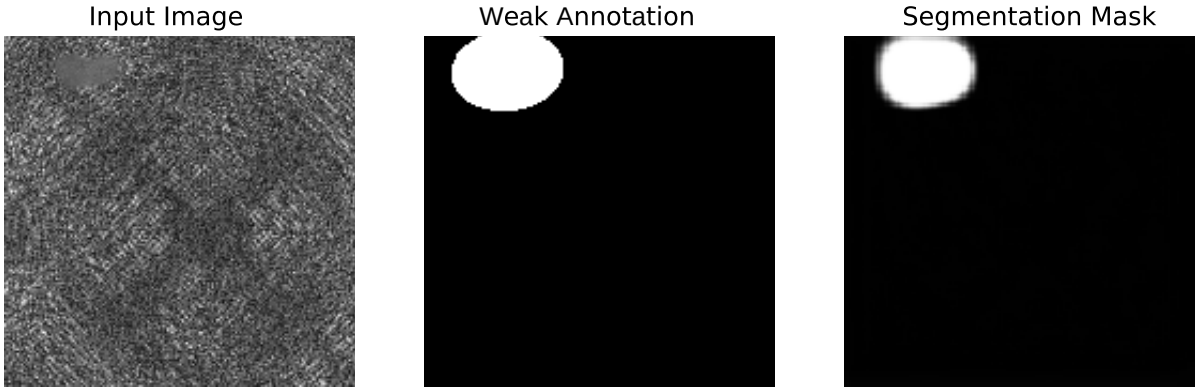

Figure 3.6: The segmentation mask shows the output of the modified CompactCNN architecture over-fitting to the DAGMC1 dataset. It fails to learn the underlying shape of the anomaly from the weak annotation. For the input image, it outputs a segmentation mask similar to the ellipse shaped annotation used for training the network.

3.3.2 Second Stage: Visualization Studies

The results of the intermediate layer visualization and activation maximization studies are discussed below.

1. **Visualization of intermediate layer outputs:** A few random samples of the intermediate layer activation study conducted for the model trained on the CrackForest dataset are shown in Figure 3.7. From these feature visualizations, we found that the initial layers were not extracting anything useful since most of them were black. The second observation was that most of the filters were looking for similar features in the later layers. This pointed towards the possibility that the model had a lot of redundancy and was over-fitting.

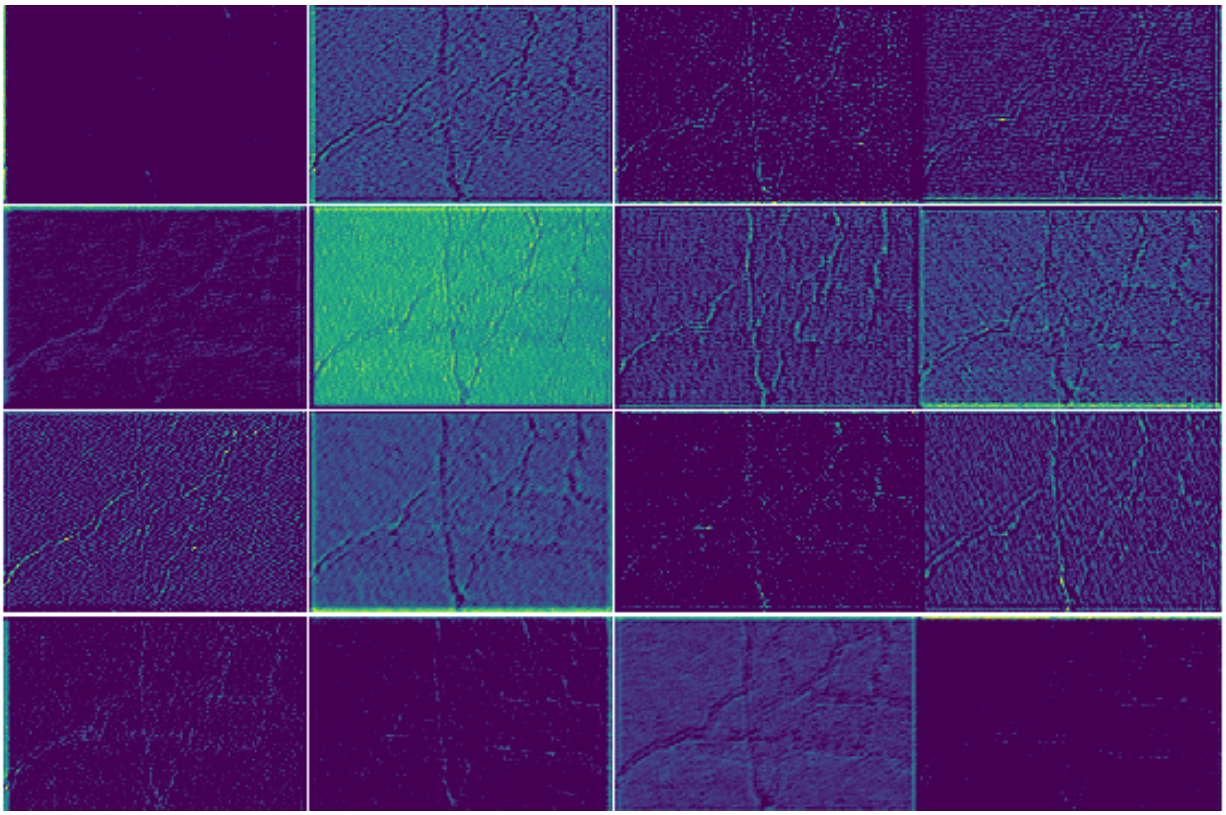

Figure 3.7: Few randomly selected samples of intermediate feature visualization for the modified CompactCNN trained on the CrackForest dataset. The key observation from these images is that most of the intermediate features extracted by the network looked similar. This pointed towards potential redundancy in the network since most of the learnt filters were looking for similar patterns. (Best viewed in colour.)

2. **Activation Maximization Study:** In this study, the activation maximization was

done for 500 steps for every filter of the modified CompactCNN model trained on the CrackForest dataset. It is important to note here that the activation maximization results are not unique. The process was conducted ten times for every kernel with step sizes of 50 and 100 and the results lead to the same observation. The preferred stimuli for the kernels looked like cracks which showed that these were looking for the right kind of inputs. A few examples of visualizations obtained using the activation maximization approach are shown in Figure 3.8. An analysis of the preferred stimuli for all the filters in the model showed that most of the filters were looking for similar patterns which were rotated by some random angle. This observation caused us to conclude that the network was possibly over-fitting and there was redundancy in the network. Because of this observation, we also hypothesized that making the initial filters rotation invariant could be potentially helpful for the anomaly detection task. This led to the inclusion of the filter bank to replace the preliminary network layers.

After these visualization studies, to validate that the network was over-fitting, we proceeded with the ablation studies, the results of which are discussed below.

3.3.3 Third Stage: Ablation Studies

The results of the ablation experiments for all the nine configurations along with the CompactCNN architecture are shown in Figure 3.9. There are 4 graphs in total which capture the F1 score and AUROC values for all the datasets for every configuration. The F1 score graphs are shown in Figure 3.9 (a) and 3.9 (b) and while the AUROC values are in Figures 3.9 (c) and 3.9 (d) respectively. For every dataset, one random sample from the validation set was chosen to show sample segmentation outputs. Figures 3.10 and 3.11 show the segmentation output of the networks after the first and twenty-fifth epoch respectively. As can be seen in Figure 3.10, for all the configurations across all the datasets, the models failed to output meaningful segmentation masks after the first epoch. This is in concurrence with the lower F1 score and AUROC values of the graphs in Figure 3.9 (a) and (c) respectively. After the 25^{th} epoch, the models learnt to output meaningful segmentation masks that localized the anomaly. This can be seen by the higher metric values in Figure 3.9 (b) and (d) as well as from the sample segmentation outputs shown in Figure 3.11. Starting with Exp4, all the configurations learnt to identify the actual shape of the anomaly for the DAGMC1 dataset, which can be seen in the first row of Figure 3.11. This confirmed our hypothesis that the modified CompactCNN network was overparameterized which caused the problem of over-fitting. Exp4 configuration was found to have the best trade-off between performance and the total number of network

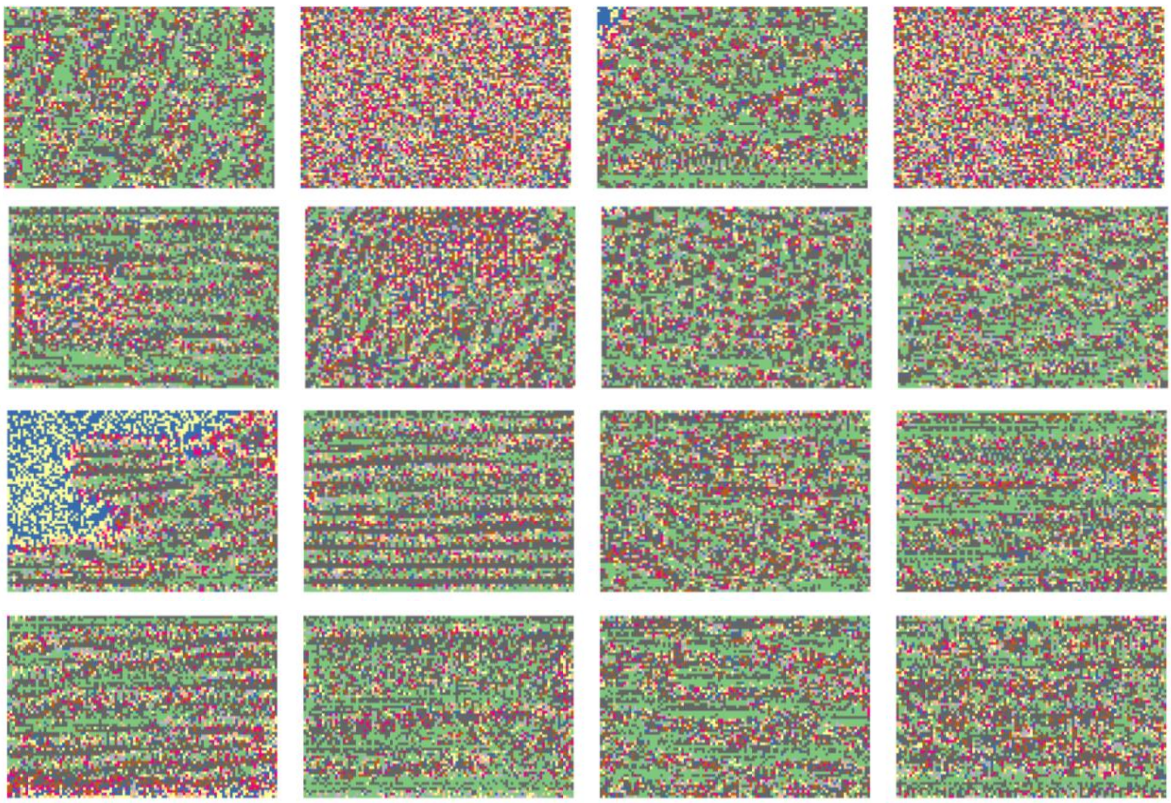

Figure 3.8: Few randomly selected activation maximization results for the modified CompactCNN trained on the CrackForest dataset. The key observations from these images were that the resultant texture looked like cracks (the anomalies in the CrackForest dataset) and all of the patterns looked similar. The green pixels indicate high intensity values and other colors indicate low intensity values. Each image shows one of the possible input patterns obtained after the gradient ascent optimization which maximised the output for a specific filter of a particular layer. Since the patterns looked like cracks, this showed that the network was looking for cracks in the images for performing the segmentation. Most of the resultant images looked similar with some random rotations. This pointed towards that the network had redundancy and was possibly over-fitting to the dataset. The noisy examples without any crack like texture are the ones that did not converge from the random initialization. (Best viewed in colour.)

parameters. It achieved a striking 94.3% reduction from 1.13 million network parameters to only 64 thousand, in comparison to the CompactCNN. Additionally, on an average across all the five datasets, it achieved a performance improvement of 51.62% to an F1 score of 0.67 and a 5.44% improvement to an AUROC value of 0.89. The Exp4 configuration was therefore selected as the AnoNet architecture. Subsequently, the AnoNet Filter Bank studies were conducted and its results are discussed in the next subsection.

3.3.4 Fourth Stage: AnoNet Filter Bank Studies

The results of the twelve AnoNet Filter Bank configurations in comparison to the CompactCNN and the DeepLabv3 architectures are presented in Figure 3.12. Similar to the ablation results, the F1 score graphs are shown in Figure 3.12 (a) and 3.12 (b) and while the AUROC values are in Figures 3.12 (c) and 3.12 (d) respectively. The segmentation outputs after the 1^{st} and 25^{th} epoch are shown in Figures 3.13 and 3.14 respectively. Interestingly, as it can be seen from Figure 3.13, all the odd numbered configurations across filters learnt to output the actual shape of the anomaly just after the first epoch for all the five datasets. The same thing can be observed from the high F1 score and AUROC value of these configurations in the graphs of Figure 3.12 (a) and (c) respectively. In concurrence with our expectations, the rotationally invariant S filter bank performed better than the directional LM and RFS filter banks. This was even though the S filter bank had only 13 filters in comparison to the 48 and 38 filters of LM and RFS filter banks respectively. It is possible that the rotational invariance of the S filter bank allowed it to extract good features across varying datasets with different texture and defect types leading to its overall best performance.

To measure the overall average performance of the models across all the datasets, we used AvgF1AUROC which is calculated as follows.

1. Calculate the average of F1 scores across all the datasets for every configuration.
2. Calculate the average of AUROC values across all the datasets for every configuration.
3. Find the average of values calculated in Step 1 and Step 2, to find the AvgF1AUROC value for every configuration.

The AvgF1AUROC metric gave the average performance of the network for all the datasets by equally weighing the F1 score and AUROC values. Remarkably, after only a single epoch the SExp1 configuration achieved the highest value AvgF1AUROC of 0.884,

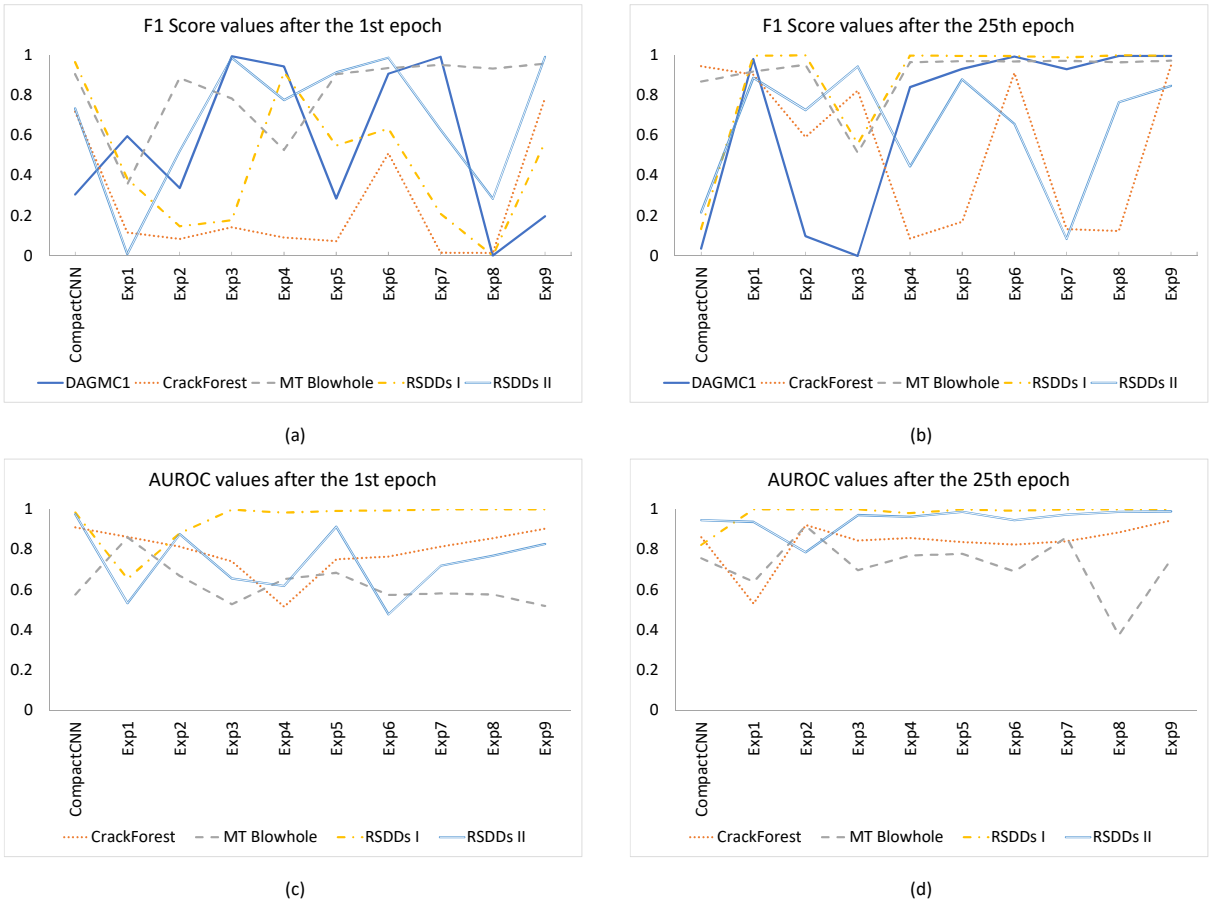

Figure 3.9: F1 score and AUROC values for all the configurations of the ablation experiments. Figures 3.9 (a) and (b) show the F1 score values for all the configurations after the first epoch and twenty-fifth epoch respectively. Figures 3.9 (c) and (d) show the AUROC values for all the configurations after the first epoch and twenty-fifth epoch respectively. As can be seen from the graphs, after the first epoch the metric values were lower in comparison to the values after the twenty-fifth epoch. (Best viewed in colour.)

followed by LMExp3 with a value of 0.8835. The configurations that had the filter layers frozen during training performed better than the ones that allowed parameter update for the filter layers. After 25 epochs, RFSExp3 had the best performance with an AvgF1AUROC value of 0.952, even though it had a poorer performance after the first epoch. The best configuration for every dataset based on F1 Score, AUROC and aver-

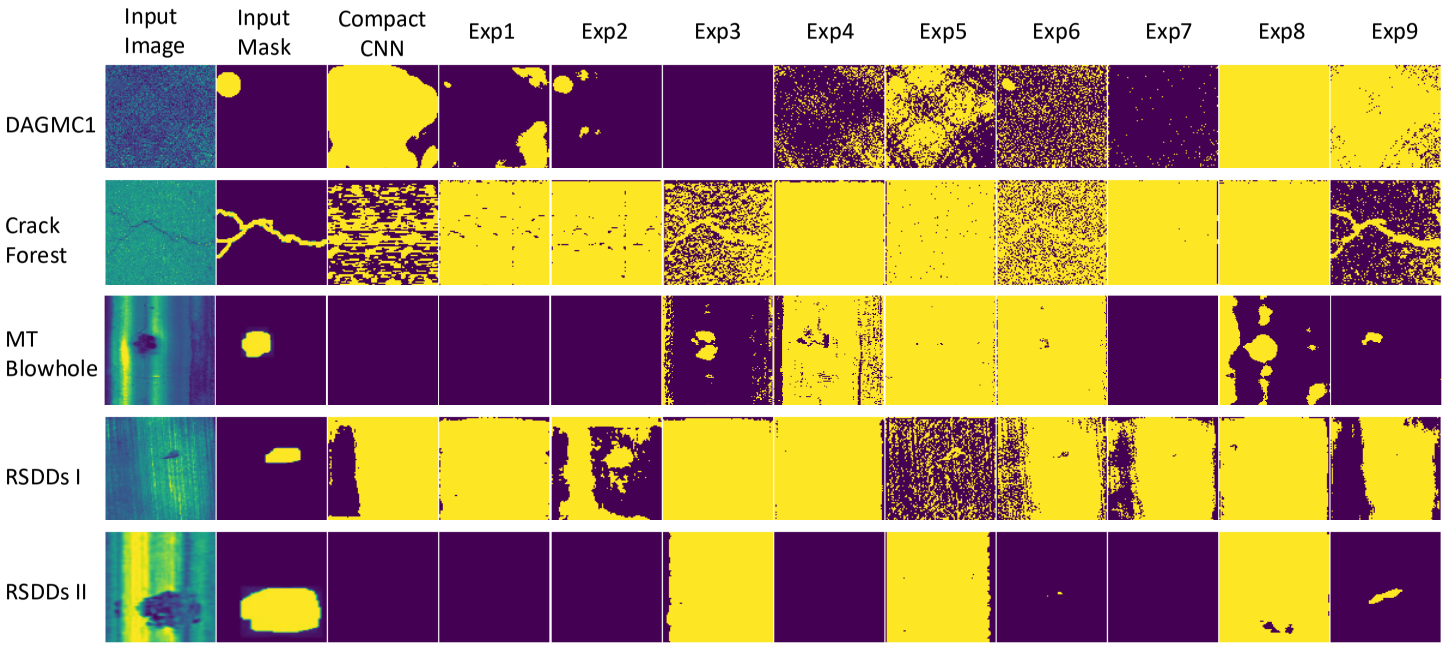

Figure 3.10: Sample segmentation outputs for the ablation experiments after the first epoch. Almost all of the configurations failed to output meaningful segmentation masks after the first epoch. (Best viewed in colour.)

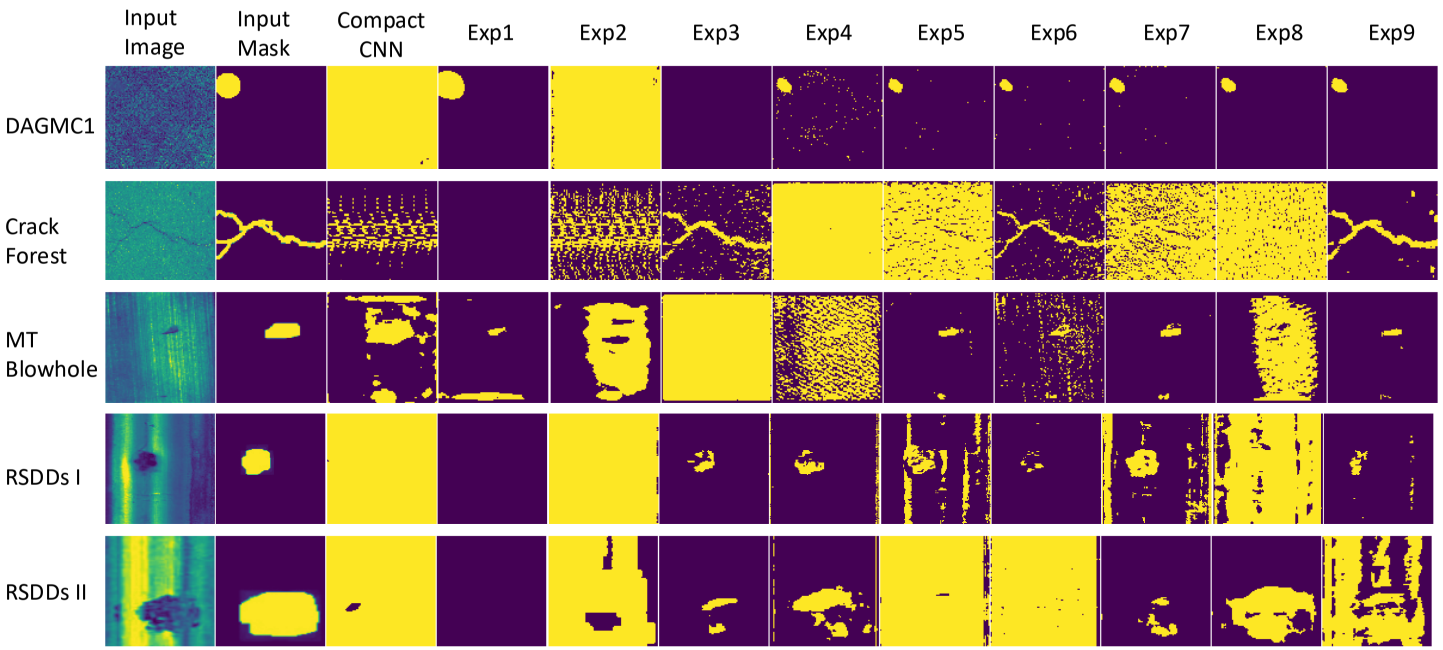

Figure 3.11: Sample segmentation outputs for the ablation experiments after the twenty-fifth epoch. The configurations learnt to output meaningful segmentation masks but the segmentation quality was poor. Interestingly, the configurations from Exp4 onward learnt to output the actual shape of the anomaly from the weakly labelled training data. (Best viewed in colour.)

Figure 3.12: F1 score and AUROC values for all the configurations of the AnoNet filter bank experiments compared to the CompactCNN and DeepLabv3. Figures 3.9 (a) and (b) show the F1 score values for all the configurations after the first epoch and twenty-fifth epoch respectively. Figures 3.9 (c) and (d) show the AUROC values for all the configurations after the first epoch and twenty-fifth epoch respectively. All the odd numbered filter bank configurations seemed to perform better than the even number configurations. The SExp1 configuration on an average performed the best across all the datasets. (Best viewed in colour.)

age of F1 Score and AUROC value after the 1^{st} epoch and the 25^{th} epoch are given in Table 3.3. We see that some of the configurations which achieved the best performance for individual datasets had their weights set to trainable. It is interesting to see that

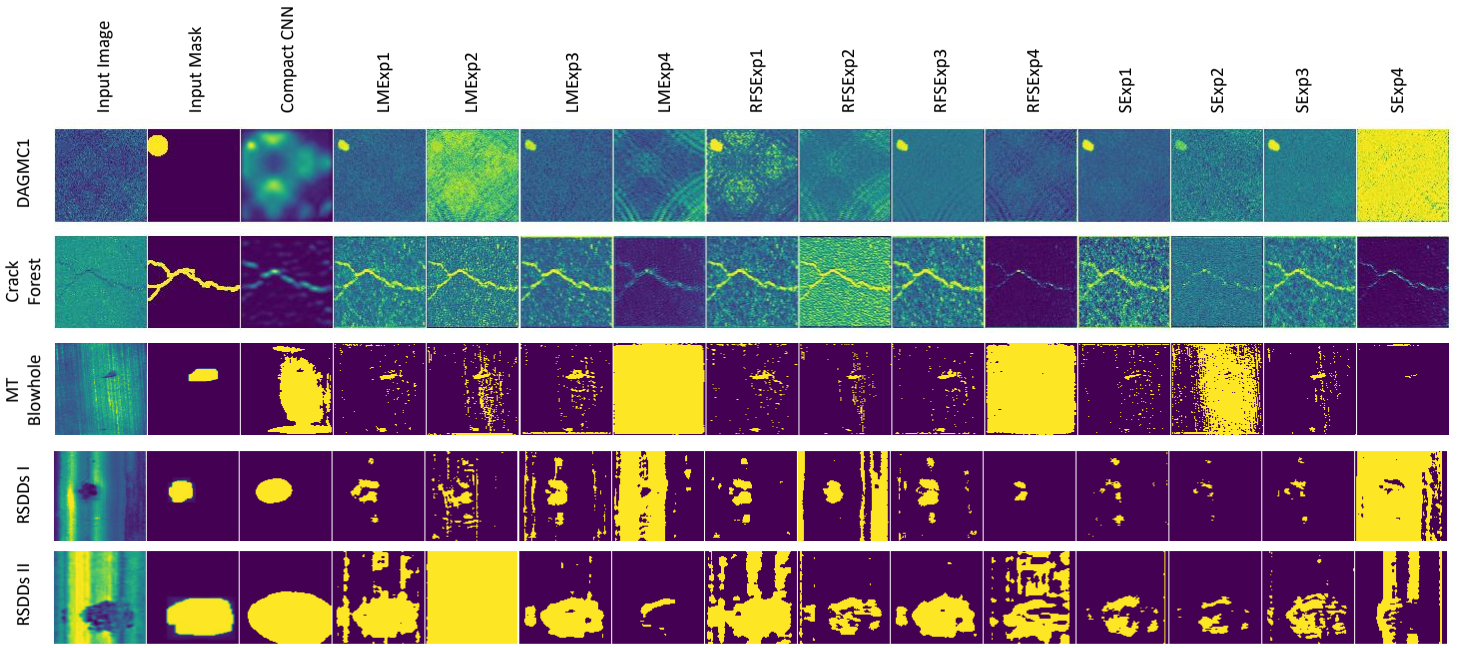

Figure 3.13: Sample segmentation outputs for filter experiments after the first epoch. As can be seen from the outputs, all the odd numbered configurations learnt to detect anomalies after the first epoch. They also learnt to detect the actual shape of the anomalies from the weak labels. (Best viewed in colour.)

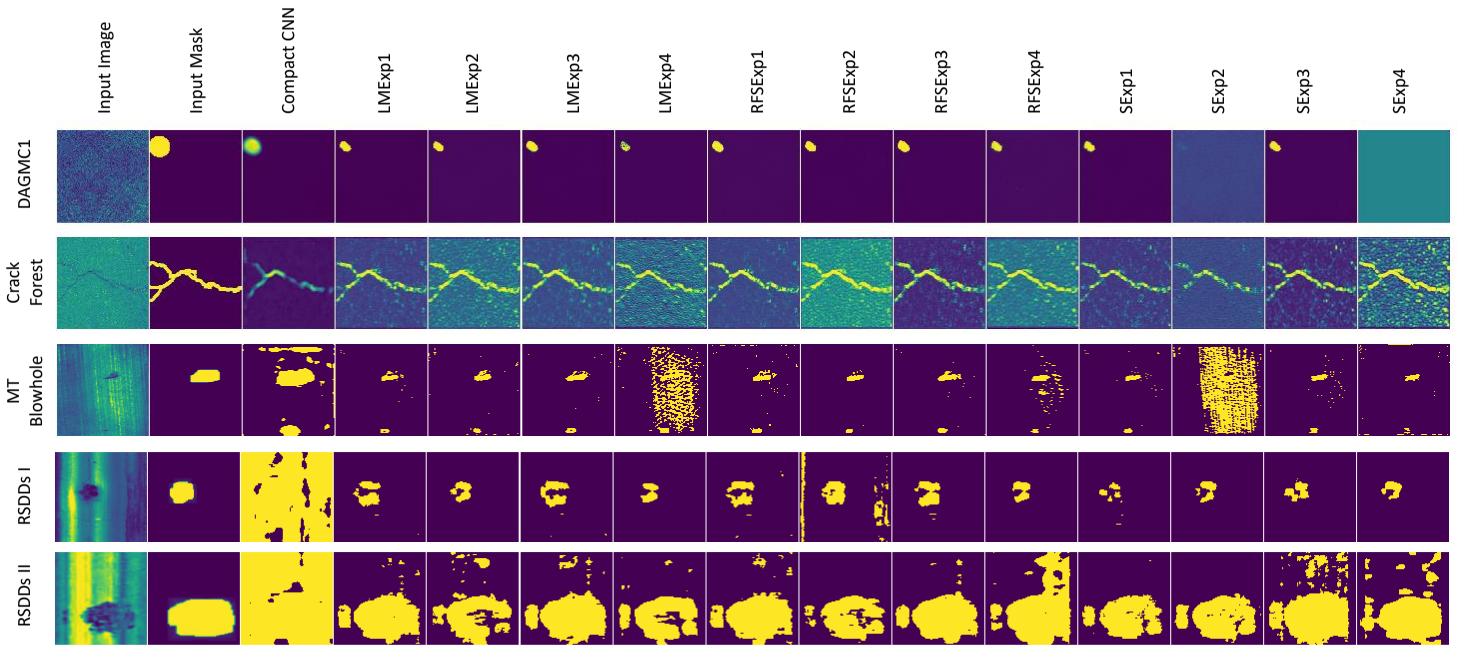

Figure 3.14: Sample segmentation outputs for filter experiments after 25 epochs. Almost all the AnoNet configurations were outputting the actual shape of the anomalies. The separation of the anomaly and normal pixels had increased for AnoNet. CompactCNN over-fit to the weak annotations and did not learn to detect the actual shape of the anomaly. (Best viewed in colour.)

even though the SExp1 configuration performed the best on an average for all datasets, it was not the best performer individually. In comparison to CompactCNN, the SExp1 configuration achieved a performance improvement of 11.5% to an AvgF1AUROC value of 0.884 after the 1^{st} epoch and a 46.4% improvement to an AvgF1AUROC value of 0.942 after the 25^{th} epoch. While the improvement in the AvgF1AUROC score in comparison to DeepLabv3 after the 1^{st} epoch was of 153.9% and 40.8% after the 25^{th} epoch respectively. Additionally, AnoNet also had a massive 92.2% reduction in the total number of parameters from 1.13 million to 64 thousand with respect to the CompactCNN. The DeepLabv3 had around 60 million parameters which is approximately 937 times more than AnoNet. The detailed performance comparison of AnoNet with CompactCNN after the 1^{st} and 25^{th} epoch for all the datasets is given in Tables 3.4 and 3.5 respectively. AnoNet performed better than CompactCNN and DeepLabv3 across all the datasets. All these performance improvements were despite that AnoNet outputted 16 times more pixel values per image in comparison to the CompactCNN. We also compared AnoNet performance with state-of-the-art techniques for Road Crack Detection. AnoNet outperforms all the methods on the CrackForest dataset which can be seen from Table 3.6.

Table 3.3: Best AnoNet configurations for every dataset based on F1 Score, AUROC value and average of F1 Score and AUROC value after the 1st and the 25th epoch.

Dataset	After 1st epoch			After 25th epoch		
	F1 Score	AUROC	Average	F1 Score	AUROC	Average
DAGMC1	RFSExp4	N.A.	N.A.	LMExp1	N.A.	N.A.
CrackForest	RFSExp4	LMExp2	LMExp2	SExp1	LMExp3	LMExp3
MT Blowhole	SExp4	LMExp3	LMExp2	LMExp3	LMExp2	LMExp2
RSDDs I	RFSExp1	LMExp1	LMExp1	RFSExp3	SExp3	SExp3
RSDDs II	RFSExp3	RFSExp3	RFSExp3	RFSExp2	RFSExp3	RFSExp3

Finally, to analyse how the choice of loss function impacts the network performance, experiments were conducted. Preliminary results from experiments conducted using the ablation experiment configurations on the CrackForest dataset using CrossEntropy as defined by Equation 3.7 and mean squared error (MSE) (defined by the Equation 3.5) as the two loss functions, show that the MSE loss worked better than CrossEntropy. In comparison to the models trained using the CrossEntropy loss, the models trained using MSE loss, on an average, achieved a 44.5% higher F1 score value of 0.71 and 6.88% higher AUROC

Table 3.4: Comparison of AnoNet, CompactCNN and DeepLabv3 after the 1st Epoch.

Dataset	AnoNet (Proposed Method)		CompactCNN		DeepLabv3	
	F1 Score	AUROC	F1 Score	AUROC	F1 Score	AUROC
DAGMC1	**0.991**	N.A.	0.305	N.A.	0.207	N.A.
CrackForest	**0.973**	**0.945**	0.717	0.910	0.020	0.436
MT Blowhole	**0.958**	**0.883**	0.904	0.576	0.022	0.491
RSDDs I	**0.964**	**0.984**	0.939	0.969	0.027	0.465
RSDDs II	**0.944**	**0.976**	0.734	0.947	0.478	0.791

Table 3.5: Comparison of AnoNet, CompactCNN and DeepLabv3 after the 25th Epoch.

Dataset	AnoNet (Proposed Method)		CompactCNN		DeepLabv3	
	F1 Score	AUROC	F1 Score	AUROC	F1 Score	AUROC
DAGMC1	**0.995**	N.A.	0.036	N.A.	0.315	N.A.
CrackForest	**0.964**	**0.956**	0.944	0.861	0.338	0.780
MT Blowhole	**0.977**	**0.878**	0.867	0.756	0.588	0.856
RSDDs I	**0.990**	**0.958**	0.134	0.823	0.641	0.791
RSDDs II	**0.972**	**0.977**	0.216	0.946	0.709	0.848

value of 0.85.

$$H = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)] \tag{3.7}$$

where H is the Cross Entropy, y_i is the label and $\hat{y}_i$ is the prediction for the i^{th} pixel.

Table 3.6: Comparison of AnoNet with the road crack detection systems on the CrackForest dataset.

Method	F1 Score
AnoNet (Proposed Method)	**0.9734**
Canny	0.3073
CrackTree [83]	0.7089
CrackIT [52]	0.7164
CrackForest (KNN) [64]	0.7944
CrackForest (SVM) [64]	0.8571
CrackForest (One-Class SVM) [64]	0.8377
Structred Prediction using CNNs [13]	0.9244

Chapter 4

Supervised Anomaly Detection using Transfer Learning

4.1 Methodology

The methodology followed for anomaly detection using network-based transfer learning is described in this section.

1. **Source Model Selection**: A source CNN model trained on a source dataset for the classification task is selected for the network-based transfer learning. For example, DenseNet161 trained on the ImageNet dataset.

2. **Source Model Modification**: The source model is then modified by the replacement of the last fully connected layer with a new layer having two output neurons. Softmax activation is applied to the layer to convert the neuron outputs into probabilities. Now the network is ready to be trained for the defect detection task.

3. **Target Model Transfer Learning**: This step involves the training of the modified neural network on the target dataset. Two different strategies can be used in this step and are as follows.

    i. **Fixed Feature Extractor**: It has been shown that deep learning models are good at extracting general features that are better then the traditional hand-crafted features for classification. In this case, all the pre-trained network parameter weight values are frozen during training (i.e. these perimeters won't

be updated during the optimization process). This causes the CNN model to function as a fixed feature extractor. Only the final fully connected softmax layer weights are learnt during the training stage.

ii. **Full Network Fine Tuning**: In this method parameters of the entire network or that of the last n layers (parameters frozen for the initial layers) are updated along with the softmax classifier during the optimization or training procedure. A lower learning rate is used because the pre-trained weights are good and don't need to be changed too fast and too much.

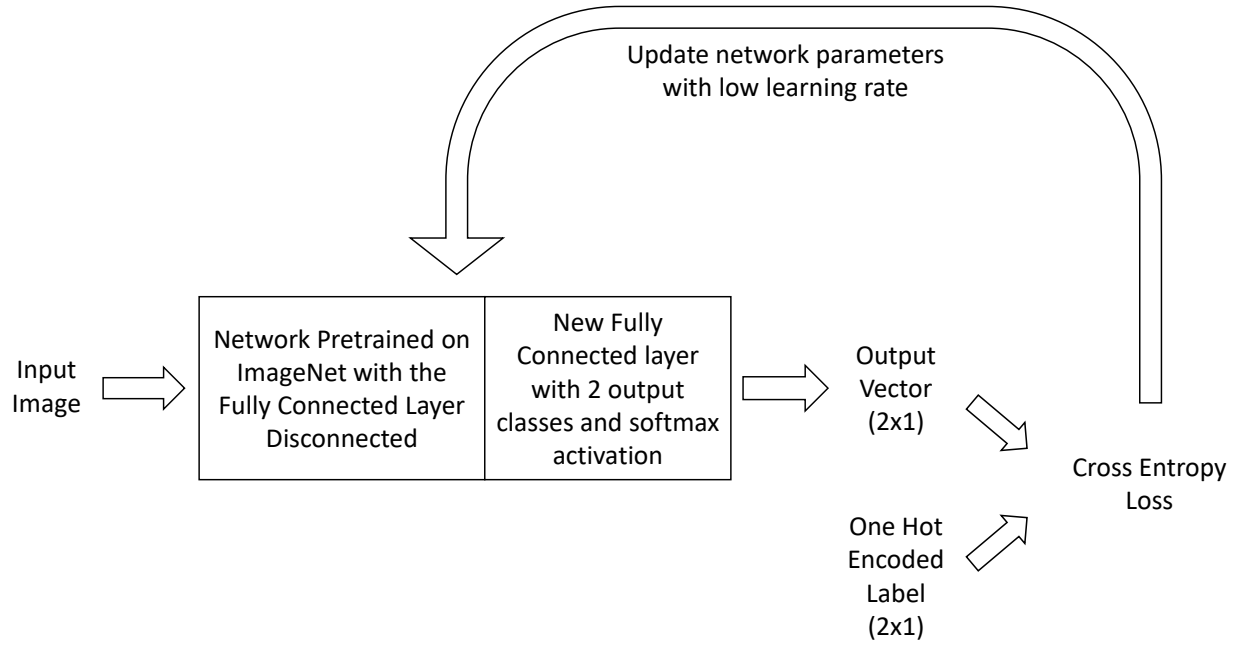

Figure 4.1: Defect Detection using network-based transfer learning. A model pre-trained on some source dataset (e.g. ImageNet) is selected as the base network. The final layers of the network are modified to have two output classes, after which the softmax activation is applied to convert the neuron outputs into probabilities. The network is then trained on the target dataset with a much smaller learning rate (e.g. 10^{-4}) to adapt it to the new dataset. The output from the anomaly class neuron is then used as an anomaly score for the sample. A high value indicates that the network is confident that the sample is anomalous.

4.2 Experiments

In this section, the overall experimental setup including the datasets, CNN architectures, implementation, training, and evaluation criteria are explained.

4.2.1 Datasets

The datasets used for the experiments are described below.

1. **The German Asphalt Pavement Distress (GAPs) v2** [66]: GAPs dataset is a high-quality dataset for pavement distress detection with damage classes as cracks, potholes, inlaid patches, applied patches, open joints and bleeding. The v2 of the dataset has 50k subset available for deep learning approaches. It contains 30k normal patches and 20k patches with defects with a patch size of 256×256 for the training set. And for the testing set, there are 6k normal patches and 4k patches with defects.

2. **DAGM dataset**[44]: It is a synthetic dataset for weakly supervised learning for industrial optical inspection. The dataset contains ten classes of artificially generated textures with anomalies. For this study, the Class 1 having the smudge defect was selected, since it presented with the maximum intra-class variance of the background texture. It (hereafter referred to as DAGMC1) contains 150 images with one defect per image and 1000 defect-free images.

3. **Magnetic Tile Defects dataset** [25]: This dataset contains images of magnetic tiles collected under varying lighting conditions. Magnetic tiles are used in engines for providing constant magnetic potential. There are five different defect types available namely Blowhole, Crack, Fray, Break and Uneven. In the experiments in addition to testing the individual defect classes, an MT_Defect category consisting of all the defect types was also created and considered.

4. **Concrete Crack** [12]: The dataset contains images of concrete with two classes namely positive (with the crack defect) and negative (without crack). There are $20,000$ 277×277 color images for each class. Images have variance in terms of surface finish and illumination conditions which makes the dataset challenging.

4.2.2 CNN architectures

The following architectures were selected for conducting the experiments. Within each category, the model configuration which achieved the lowest error on the ImageNet dataset was selected.

1. **DenseNet** Densely Connected Convolutional Networks [24] (DenseNets) introduced the concept of inputs from every preceding layer in the dense blocks. Every layer is connected to every other layer in a feed forward fashion so that the network with L layers has $\frac{L(L+1)}{2}$ direct connections. DenseNet-161 architecture was used as the source network for the experiments.

2. **ResNet** Deep Residual Networks [22] introduced the concept of identity shortcut connections that skip one or more layers. These were introduced in 2015 by Kaiming He. et.al. and bagged 1^{st} place in the ILSVRC 2015 classification competition . ResNet-152 architecture is used for the experiments.

3. **VGGNet** VGGnet was invented by the Visual Geometry Group from the University of Oxford. It introduced the use of successive layers of 3×3 filters instead of large-size filters such as 11×11 and 7×7. VGG19 was chosen for the experiments.

4.2.3 Implementation

PyTorch [54] version 1.3 was used for conducting all the experiments. Publicly available implementations of the selected models were used from the torchvision package version 0.2.2. Model weights pre-trained on ImageNet dataset available in the PyTorch model zoo were used for the experiments. Adam [29] optimizer with default settings was used. The learning rate was set to 10^{-4}. All the experiments were conducted for 25 epochs. The input images were resized to $224 \times 224 \times 3$ before feeding to the network because of the fully connected layers. The prediction output from the anomaly/defect neuron was used as the anomaly score and also for performing the classification. The loss function used was CrossEntropy which is defined by equation 3.7. The evaluation metrics used were F1 Score and AUROC (subsection 3.2.7).

4.3 Results

Figure 4.2 summarises the results of all the experiments conducted for the various dataset and CNN architecture configurations. Figures 4.2 (a), (b) and (c) show the AUROC and F1 Score values for the Fixed Feature Extractor and Full Network Fine Tuning experiments for DenseNet161, ResNet152 and Vgg19 respectively. The values shown are for the best model per architecture and dataset based on the lowest validation loss. It is important to note that for calculating the F1 scores a threshold value of 0.5 was used since that is the mean value of the output range of the neuron with softmax activation applied to it. The F1 score value will vary depending on the choice of threshold. But the AUROC score takes into account all the possible threshold values in its calculation. One clear observation from all the experiments is that on an average, across all the dataset and CNN architecture configurations Full Network Fine Tuning worked better than the Fixed Feature Extractor approach. This showed that the initial layers which are often attributed to be good at extracting general features, also need to be trained while performing the network-based transfer learning. Fine tuning the network weights with a lower learning rate in comparison to the learning rate used during the training on the source dataset leads to weights that better optimize the cost function for the target task and dataset.

On average across all the datasets, using the Full Network Fine Tuning approach the Vgg19 architecture performed the best with F1 Score and AUROC values of 0.8914 and 0.9766 respectively. In the fixed feature extractor approach too Vgg19 performed the best on an average across all the datasets but the F1 Score and AUROC values were lower by 49% and 28% respectively. DAGMC1 was the only synthetic dataset in the experiments and as expected all the three architectures are perfectly able to separate the defects or anomalies from the normal samples. On the extremely challenging GAPSv2 dataset DenseNet161 performed the best with F1 Score and AUROC values of 0.9882 and 0.9979 respectively. ConcreteCrack dataset is the only dataset on which on average the fixed feature extractor approach performed better than the full network fine tuning. However, the performance gap was marginal in comparison to other datasets. It was 2% for the F1 Score and 4% for the AUROC value. On the magnetic tile dataset (datasets with the prefix MT) as expected average of the best models trained for single defect category outperformed the best model trained on the mixture of all the defects. The improvement for F1 Score and AUROC values was that of 6% and 4% respectively. Another thing to note is that the output of the anomaly/defect neuron being used as an anomaly score worked well. It resulted in a very high separating power of the networks between the anomalous and normal samples. This is evident from the impressive average AUROC value of 0.9766 as mentioned earlier in this section.

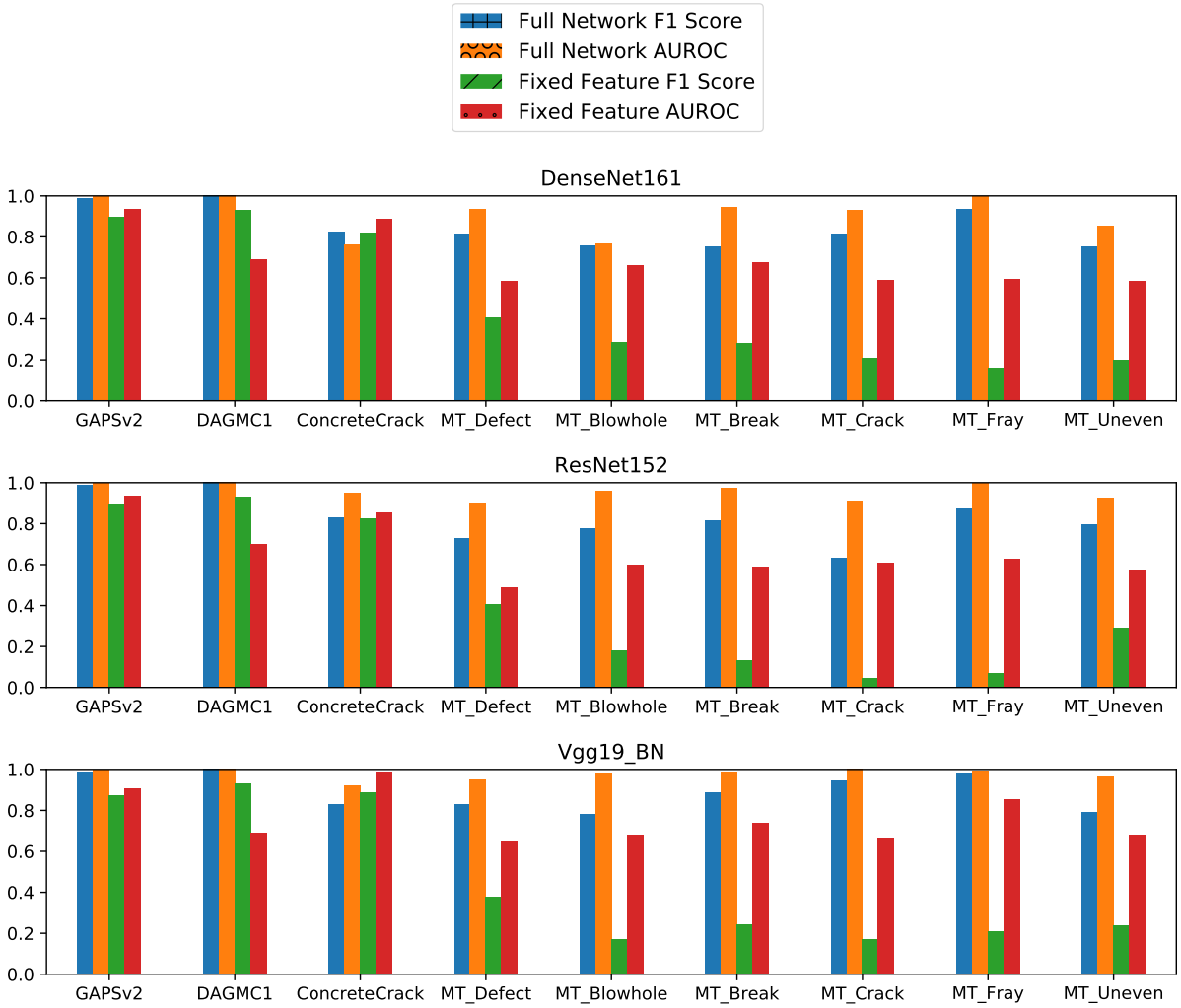

Figure 4.2: Results of the experiments conducted on all the datasets and CNN architectures. Figures 4.2 (a), (b) and (c) show the AUROC and F1 Score values for the Fixed Feature Extractor and Full Network Fine Tuning experiments for DenseNet161, ResNet152 and Vgg19 respectively. The values shown are for the best model per architecture and dataset based on the lowest validation loss. It can be observed across the datasets and the architectures, that on an average the full network fine tuning seems to work better than the fixed feature extractor approach. (Best viewed in colour.)

Chapter 5

Conclusion

Anomaly detection in textured surfaces is an interesting and relevant problem with practical applications having huge financial implications specifically in industrial defect detection and infrastructure asset inspection and maintenance. As highlighted in the introduction, limited dataset size and a low number of anomalous instances are amongst the most important challenges in anomaly detection. Also, what an anomaly is, varies from domain to domain, which adds to the complexity. While unsupervised and semi-supervised approaches that try to tackle these challenges do exist, they suffer from poor performance compared to the supervised techniques. We have explored and proposed two approaches for anomaly detection that directly address these key pain points. The two approaches are "AnoNet: Weakly Supervised Anomaly Detection in Textured Surfaces" and "Supervised Anomaly Detection using Transfer Learning".

Weak annotation eases the time and human labor-intensive task of generating pixel-level annotated datasets by replacing them with coarse annotations. However, this makes the anomaly detection problem even more complicated because a lot of pixels in the training data will have a wrong label. The Garbage In Garbage Out rule applies for model-based approaches and especially for deep learning methods. We have developed the fully convolutional AnoNet architecture (Figure 3.1) for anomaly detection in textured surfaces using weakly labelled data. It uses a unique filter bank based initialization technique which leads to faster training. For a $H \times W \times 1$ input image, it outputs a $H \times W \times 1$ segmentation mask. This prevents the loss of spatial localization of the anomaly. The network has the valuable ability to learn to output the real shape of the anomaly despite the weak annotations. AnoNet is compact with only 64 thousand parameters. Not only does this result in the reduction of the computational complexity of the model leading to faster inference time, but it also overcomes the challenge of over-fitting, by design. To the best

of our knowledge, no such work has been done for weakly supervised anomaly detection in textured surfaces. Comprehensive experiments conducted on four challenging datasets showed that, compared to CompactCNN and DeepLabv3, AnoNet achieved an impressive improvement in performance on an average across all datasets by 106% to an F1 score of 0.98 and by 13% to an AUROC value of 0.942. This performance improvement was even though AnoNet predicted 16 times more pixels per image in comparison to CompactCNN. The model learnt to detect anomalies after just a single epoch. AnoNet has the advantage that it can learn from a few images and generalises well to similar anomaly detection tasks. Currently, there is no bench-marking available for weakly supervised anomaly detection.

We also investigated network-based transfer learning using CNNs for anomaly detection, which overcame the challenge of training from a limited number of anomalous samples. The method achieved impressive F1 Score and AUROC values of 0.8914 and 0.9766 respectively, on an average across four challenging datasets. Within network-based transfer learning, we explored Fixed Feature Extraction and Full Network Fine Tuning approaches. Results showed that the full network fine-tuning approach worked better than the fixed feature extraction approach. The use of the output value from the neuron responsible for the anomaly (defect) class as an anomaly score led to excellent AUROC values showing the strong separation capability of the CNNs across all the datasets.

For future work on AnoNet, investigations need to be conducted on how the choice of filter banks and the trainable parameter for the filter bank layer affects the performance of AnoNet on different types of textures and anomalies. Since the ground truth itself is not accurate in weakly labelled anomaly detection, the Intersection over Union (IoU) metric is another possible way for measuring the quantitative performance. Additionally, it would be interesting to see whether AnoNet achieves similar performance on datasets with more than one defect type per image. Next, for network-based transfer learning, how the choice of the activation function of the final classifier affects performance could be explored. More CNN architectures could be analysed to see how the choice of the architecture affects the performance for different defect types.

In this research, we have successfully explored and developed two approaches for supervised anomaly detection using deep learning with promising results on several challenging real-world datasets. This showed that the developed methods are generalisable to anomaly detection in textured surfaces and are not limited to any specific type of texture or anomaly. However, is anomaly detection a solved problem? There is research potential in semi-supervised and unsupervised anomaly detection. With the recent advancements in deep learning generative models such as generative adversarial networks and variational auto-encoders, the performance gap compared to supervised learning could potentially be bridged.

References

[1] D. Ai, G. Jiang, L. Siew Kei, and C. Li. Automatic pixel-level pavement crack detection using information of multi-scale neighborhoods. *IEEE Access*, 6:24452–24463, 2018.

[2] S. Akcay, A. A. Abarghouei, and T. P. Breckon. GANomaly: Semi-supervised anomaly detection via adversarial training. *CoRR*, abs/1805.06725, 2018.

[3] G. Antipov, S.-A. Berrani, N. Ruchaud, and J.-L. Dugelay. Learned vs. hand-crafted features for pedestrian gender recognition. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, pages 1263–1266, New York, NY, USA, 2015. ACM.

[4] B. Barz, E. Rodner, Y. G. Garcia, and J. Denzler. Detecting regions of maximal divergence for spatio-temporal anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(5):1088–1101, May 2019.

[5] C. Chahla, H. Snoussi, F. Abdallah, and F. Dornaika. Learned versus handcrafted features for person re-identification. *International Journal of Pattern Recognition and Artificial Intelligence*, 34(4):2055009 (19 pages), May 2019.

[6] R. Chalapathy and S. Chawla. Deep learning for anomaly detection: A survey. *CoRR*, abs/1901.03407, 2019.

[7] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):15, 2009.

[8] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *ArXiv*, abs/1706.05587, 2017.

[9] Chi-Ho Chan and G. K. H. Pang. Fabric defect detection by fourier analysis. *IEEE Transactions on Industry Applications*, 36(5):1267–1276, Sep. 2000.

[10] M. Cimpoi, S. Maji, and A. Vedaldi. Deep filter banks for texture recognition and segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3828–3836, June 2015.

[11] S. Faghih-Roohi, S. Hajizadeh, A. Nez, R. Babuska, and B. De Schutter. Deep convolutional neural networks for detection of rail surface defects. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 2584–2589, July 2016.

[12] R. Fan, M. J. Bocus, Y. Zhu, J. Jiao, L. Wang, F. Ma, S. Cheng, and M. Liu. Road crack detection using deep convolutional neural network and adaptive thresholding. *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 474–479, 2019.

[13] Z. Fan, Y. Wu, J. Lu, and W. Li. Automatic pavement crack detection based on structured prediction with the convolutional neural network. *ArXiv*, abs/1802.02208, 2018.

[14] C. Feng, H. Zhang, S. Wang, Y. Li, H. Wang, and F. Yan. Structural damage detection using deep convolutional neural network and transfer learning. *KSCE Journal of Civil Engineering*, 23(10):4493–4502, Oct 2019.

[15] J. Gan, Q. Li, J. Wang, and H. Yu. A hierarchical extractor-based visual rail surface inspection system. *IEEE Sensors Journal*, 17(23):7935–7944, Dec 2017.

[16] J. Gan, Q. Li, J. Wang, and H. Yu. A hierarchical extractor-based visual rail surface inspection system. *IEEE Sensors Journal*, 17(23):7935–7944, Dec 2017.

[17] J. . Geusebroek, A. W. M. Smeulders, and J. van de Weijer. Fast anisotropic gauss filtering. *IEEE Transactions on Image Processing*, 12(8):938–943, Aug 2003.

[18] M. Goldstein and S. Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PLOS ONE*, 11(4):1–31, 04 2016.

[19] M. Haindl and J. Filip. *Motivation*, pages 1–7. Springer London, London, 2013.

[20] R. M. Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804, May 1979.

[21] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 1026–1034, Washington, DC, USA, 2015. IEEE Computer Society.

[22] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[23] M. D. Hssayeni, S. Saxena, R. Ptucha, and A. Savakis. Distracted driver detection: Deep learning vs handcrafted features. *Electronic Imaging*, 2017(10):20–26, 2017.

[24] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017.

[25] Y. Huang, C. Qiu, and K. Yuan. Surface defect saliency of magnetic tile. *The Visual Computer*, Aug 2018.

[26] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 448–456. JMLR.org, 2015.

[27] L. P. K. Kenji Kawaguchi and Y. Bengio. Generalization in deep learning. In *Mathematics of Deep Learning, Cambridge University Press, to appear. Prepint avaliable as: MIT-CSAIL-TR-2018-014, Massachusetts Institute of Technology*, 2018.

[28] A. Kensert, P. J. Harrison, and O. Spjuth. Transfer learning with deep convolutional neural networks for classifying cellular morphological changes. *SLAS DISCOVERY: Advancing Life Sciences R&D*, 24(4):466–475, 2019. PMID: 30641024.

[29] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[30] A. Krizhevsky, V. Nair, and G. Hinton. Cifar-10 (canadian institute for advanced research).

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017.

[32] A. Kumar. Computer-vision-based fabric defect detection: A survey. *IEEE Transactions on Industrial Electronics*, 55(1):348–363, Jan 2008.

[33] A. Kumar and G. K. H. Pang. Defect detection in textured materials using gabor filters. *IEEE Transactions on Industry Applications*, 38(2):425–440, March 2002.

[34] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, Jun 2001.

[35] Q. Li and S. Ren. A visual detection system for rail surface defects. *Trans. Sys. Man Cyber Part C*, 42(6):1531–1542, Nov. 2012.

[36] P. Lillian, R. Meyes, and T. Meisen. Ablation of a robot's brain: Neural networks under a knife. *CoRR*, abs/1812.05687, 2018.

[37] C. X. Ling, J. Huang, and H. Zhang. Auc: A statistically consistent and more discriminating measure than accuracy. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 519–524, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.

[38] L. Liu, J. Chen, P. Fieguth, G. Zhao, R. Chellappa, and M. Pietikäinen. From bow to cnn: Two decades of texture representation for texture classification. *International Journal of Computer Vision*, 127(1):74–109, Jan 2019.

[39] C. Mandriota, M. Nitti, N. Ancona, E. Stella, and A. Distante. Filter-based feature selection for rail defect detection. *Machine Vision and Applications*, 15(4):179–185, Oct 2004.

[40] A. Z. Manik Varma. Texture classification — filter banks. http://www.robots.ox.ac.uk/~vgg/research/texclass/filters.html.

[41] M. A. Marnissi, H. Fradi, and J. Dugelay. On the discriminative power of learned vs. hand-crafted features for crowd density analysis. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2019.

[42] J. Masci, U. Meier, D. C. Ciresan, J. Schmidhuber, and G. Fricout. Steel defect classification with max-pooling convolutional neural networks. *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6, 2012.

[43] J. Masci, U. Meier, G. Fricout, and J. Schmidhuber. Multi-scale pyramidal pooling network for generic steel defect classification. In *IJCNN*, pages 1–8. IEEE, 2013.

[44] T. H. Matthias Wieler. Weakly supervised learning for industrial optical inspection. https://hci.iwr.uni-heidelberg.de/node/3616, 2007.

[45] D. Mery, V. Riffo, U. Zscherpel, G. Mondragón, I. Lillo, I. Zuccar, H. Lobel, and M. Carrasco. Gdxray: The database of x-ray images for nondestructive testing. *Journal of Nondestructive Evaluation*, 34(4):42, Nov 2015.

[46] D. T. R. Miller and D. E. Zaloshnja. On a crash course: The dangers and health costs of deficient roadways. Technical report, Pacific Institute for Research and Evaluation, 2009.

[47] L. Nanni, S. Ghidoni, and S. Brahnam. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognition*, 71:158 – 172, 2017.

[48] N. Neogi, D. K. Mohanta, and P. K. Dutta. Review of vision-based steel surface inspection systems. *EURASIP Journal on Image and Video Processing*, 2014(1):50, Nov 2014.

[49] A. Nguyen, J. Yosinski, and J. Clune. Understanding Neural Networks via Feature Visualization: A survey. *arXiv e-prints*, page arXiv:1904.08939, Apr 2019.

[50] A. S. of Civil Engineers. Infrastructure report card. https://www.infrastructurereportcard.org/cat-item/roads/, 2017.

[51] O. of Safety Analysis. Train accidents by railroad groups. https://safetydata.fra.dot.gov/OfficeofSafety/publicsite/on_the_fly_download.aspx, 2015.

[52] H. Oliveira and P. L. Correia. Crackit an image processing toolbox for crack detection and characterization. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 798–802, Oct 2014.

[53] G. W. H. Organization. Global status report on road safety 2018.

[54] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

[55] F. Pernkopf and P. O'Leary. Visual inspection of machined metallic high-precision surfaces. *EURASIP Journal on Advances in Signal Processing*, 2002(7):650750, Jul 2002.

[56] V. H. Pham and B. R. Lee. An image segmentation approach for fruit defect detection using k-means clustering and graph-based algorithm. *Vietnam Journal of Computer Science*, 2(1):25–33, Feb 2015.

[57] K. Pogorelov, O. Ostroukhova, A. Petlund, P. Halvorsen, T. de Lange, H. N. Espeland, T. Kupka, C. Griwodz, and M. Riegler. Deep learning and handcrafted feature based approaches for automatic detection of angiectasia. In *2018 IEEE EMBS International Conference on Biomedical Health Informatics (BHI)*, pages 365–368, March 2018.

[58] D. Racki, D. Tomazevic, and D. Skocaj. A compact convolutional neural network for textured surface anomaly detection. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1331–1339, March 2018.

[59] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In M. Niethammer, M. Styner, S. Aylward, H. Zhu, I. Oguz, P.-T. Yap, and D. Shen, editors, *Information Processing in Medical Imaging*, pages 146–157, Cham, 2017. Springer International Publishing.

[60] C. Schmid. Constructing models for content-based image retrieval. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 2, pages II–II, Dec 2001.

[61] O. Schwartz and E. P. Simoncelli. Natural signal statistics and sensory gain control. *Nature Neuroscience*, 4(8), August 2001.

[62] J. E. See, C. G. Drury, A. Speed, A. Williams, and N. Khalandi. The role of visual inspection in the 21st century. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 61(1):262–266, 2017.

[63] A. Serdaroglu, A. Ertuzun, and A. Ercil. Defect detection in textile fabric images using wavelet transforms and independent component analysis. *Pattern Recognition and Image Analysis*, 16(1):61–64, Jan 2006.

[64] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen. Automatic road crack detection using random structured forests. *IEEE Transactions on Intelligent Transportation Systems*, 17(12):3434–3445, 2016.

[65] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[66] R. Stricker, M. Eisenbach, M. Sesselmann, K. Debes, and H.-M. Gross. Improving visual road condition assessment by extensive experiments on the extended gaps dataset. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2019.

[67] X. Sun, J. Gu, S. Tang, and J. Li. Research progress of visual inspection technology of steel products a review. *Applied Sciences*, 8(11), 2018.

[68] H. Tamura, S. Mori, and T. Yamawaki. Textural features corresponding to visual perception. *IEEE Transactions on Systems, Man, and Cybernetics*, 8(6):460–473, June 1978.

[69] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu. A survey on deep transfer learning. In *ICANN 2018*, 2018.

[70] L. Torrey and J. W. Shavlik. Transfer learning. 2009.

[71] D.-M. Tsai and T.-Y. Huang. Automated surface inspection for statistical textures. *Image and Vision Computing*, 21(4):307 – 323, 2003.

[72] M. Varma and A. Zisserman. Texture classification: Are filter banks necessary? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 691–698, June 2003.

[73] D. Weimer, B. Scholz-Reiter, and M. Shpitalni. Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection. *CIRP Annals*, 65(1):417 – 420, 2016.

[74] X. Xie. A review of recent advances in surface defect detection using texture analysis techniques. *ELCVIA Electronic Letters on Computer Vision and Image Analysis*, 7(3):1–22, 2008.

[75] J. Yosinski, J. Clune, A. M. Nguyen, T. J. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *CoRR*, abs/1506.06579, 2015.

[76] S. Youkachen, M. Ruchanurucks, T. Phatrapomnant, and H. Kaneko. Defect segmentation of hot-rolled steel strip surface by using convolutional auto-encoder and conventional image processing. In *2019 10th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES)*, pages 1–5, March 2019.

[77] H. Yu, Q. Li, Y. Tan, J. Gan, J. Wang, Y. Geng, and L. Jia. A coarse-to-fine model for rail surface defect detection. *IEEE Transactions on Instrumentation and Measurement*, 68(3):656–666, March 2019.

[78] M. R. Zare, D. O. Alebiosu, and S. L. Lee. Comparison of handcrafted features and deep learning in classification of medical x-ray images. In *2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP)*, pages 1–5, March 2018.

[79] M. D. Zeiler. Adadelta: An adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[80] H. Zenati, M. Romain, C. S. Foo, B. Lecouat, and V. R. Chandrasekhar. Adversarially learned anomaly detection. *2018 IEEE International Conference on Data Mining (ICDM)*, pages 727–736, 2018.

[81] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. 2017.

[82] Z.-H. Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53, 08 2017.

[83] Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang. Cracktree: Automatic crack detection from pavement images. *Pattern Recognition Letters*, 33(3):227 – 238, 2012.

[84] A. EKER and A. G. YKSEK. Stacked autoencoder method for fabric defect detection. *Cumhuriyet niversitesi Fen Edebiyat Fakltesi Fen Bilimleri Dergisi*, 38:342 – 354, 2017.