An Adaptive Human-in-the-loop Approach to Continuous Understanding of Additive Manufacturing Processes with Computer Vision

Xiao Liu, M.Eng.

Supervised by Dr. Alessandra Mileo

and Prof. Alan F. Smeaton

A thesis presented for the degree of Doctor of Philosophy

School of Computing

Dublin City University

August 2024

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work

Xiao Liu

Xiao Liu

ID No.: 56101686

14/August/2024

Dedication

To all the people who are kind and friendly to me.

To the part of me that said, “I still have the will to fight.”

Acknowledgements

I would like to thank my joint supervisor Dr. Alessandra Mileo for her patience and help on my research over the last five years as well as her kind support during the lockdown. I would also like to thank my other supervisor Professor Alan Smeaton for his patience and tireless support to make this thesis complete and on time. I must thank him again for the encouragement and the inspiration from his passion in work and life. I would like to thank my independent panel member Prof. Dermot Brabazon for his support and kind communications from which I gained more understanding of academic life.

I am also indebted to my late additional supervisor Dr. Kevin McGuinness for his professional direction and support when I was new to this research area. My sincere condolences on his passing.

I would also like express my deepest appreciation to my family. They are always my reason to survive on such a long journey.

Finally, I would like to acknowledge support from Science Foundation Ireland (SFI) under Grant Number 16/RC/3872 which is co-funded under the European Regional Development Fund and by I-Form industry partners.

Contents

List of Figures

List of Tables

Glossary

Adam Adaptive Moment Estimation .

AIFT Active Incremental Fine-Tuning.

AM Additive Manufacturing .

CNN Convolutional Neural Network.

DAGM Deutsche Arbeitsgemeinschaft für Mustererkennung.

DoG Difference of Gaussian.

DoH Determinant of Hessian.

HITL Human-In-The-Loop.

HOG Histogram of Oriented Gradients.

LoG Laplacian of Gaussian.

MQS Membership Query Synthesis.

PBF Powder Bed Fusion.

PBS Pool-Based sampling.

QBC Query-By-Committee.

R-CNN Region with Convolutional Neural Network.

RMSprop Root Mean Square Propagation.

ROI Regions of Interest.

ROS Random Oversampling.

RPNs Region Proposal Networks.

RUS Random Undersampling.

SBSS Stream-Based Selective Sampling.

SGD Stochastic Gradient Descent.

SIFT	Scale Invariant and Feature Transform.
SLS	Selective Laser Sintering.
SMOTE	Synthetic Minority Oversampling TEchnique .
SVM	Support Vector Machine.
TNR	True Negative Rate.
TPR	True Positive Rate.

Publications from this thesis

1. Liu, X., Mileo, A and Smeaton, A. F. (in preparation). The Small Data Challenge in Additive Manufacturing Defect Detection Using Active Learning and Auto-Labelling. Additive Manufacturing Journal.

2. Liu, X., Mileo, A and Smeaton, A. F. A Systematic Review of Available Datasets for Machine Learning-Based Fault Detection in Additive Manufacturing. IEEE Access (under review).

3. Liu, X., Mileo, A. and Smeaton, A. F. (2023). Defect Classification in Additive Manufacturing Using CNN-Based Vision Processing. In *25th Irish Machine Vision and Image Processing Conference*, University of Galway, Ireland. `https://doi.org/10.5281/zenodo.8230600`

4. Liu, X., Smeaton, A. F., and Mileo, A. (2022). An adaptive human-in-the-loop approach to emission detection of additive manufacturing processes and active learning with computer vision. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 4021–4027. IEEE Computer Society. `http://doi.org/10.1109/BigData55660.2022.10020525`

5. Liu, X. (2022) Emission Images Defect Samples. Dataset. Figshare. `https://doi.org/10.6084/m9.figshare.21280104.v1`

6. Liu, X. and Mileo, A. (2021). A deep learning approach to defect detection in additive manufacturing of titanium alloys. In *2021 International Solid Freeform Fabrication Symposium.* University of Texas at Austin. `http://dx.doi.org/10.26153/tsw/17564`

Abstract

An adaptive human-in-the-loop approach to continuous understanding of Additive Manufacturing processes with computer vision Xiao Liu

In Additive Manufacturing (AM), recent developments in in-situ monitoring and process control allow the collection of large amounts of emissions data during the build and modification process of the parts being manufactured. This data then can be used as source to further construct 2D and 3D representations of the printed parts. However, the inspection, labeling and analysis as well as the characterisation of this data still remains a manual process. The aim of this research is to determine if and how Machine Learning techniques can automatically inspect and annotate this generated data, thereby reducing manual workload and associated costs. More specifically, this work will look at two scenarios: firstly, using convolutional neural networks (CNNs) to inspect and classify the data collected by in-situ monitoring and secondly, applying Active Learning and Semi-supervised learning to accelerate the data labeling process while continuously gaining understanding about the manufactured object from the data generated during the AM process. Ultimately this work could be used to help with decisions made by an AM operator during the AM process allowing modification of the output during the actual manufacture process.

Chapter 1

Introduction

Additive manufacturing (AM), also known as 3D printing, is an important manufacturing technology that has emerged at a level of scale only very recently. According to [1] it offers multiple advantages including on-demand manufacturing, thus making it cost-effective, manufacturing which can be personalised thus making it flexible and even sustainable precisely because manufacture is done only when needed. AM operates by using a laser which scans and burns or melts some input material and a melt-pool is created at the laser-material interaction point. This forms a layer of the overall object or part being manufactured, and the process continues layer by layer from the bottom up, until the manufacture of the overall object or part is complete.

The formation of defects in the resulting manufactured parts such as tensile weakness, is typically related to the stability of the melt-pool during manufacture. Due to thermal instability, the melt-pool can create different levels of emissions. A more unstable and volatile melt-pool will typically emit a greater number of emissions [2], leading to the formation of defects in the resulting part, and this is a serious problem in the more widespread use of AM [3].

Recent developments in monitoring and process control in the AM process itself have resulted in a significant enhancement of the quality of the AM process and reduced the amount of inter-build variation and interruption in material manufacturing [4]. Furthermore, given recent advances in computer vision and the availability of potentially large amounts of data collected from in-situ monitoring of emissions from melt-pools during additive manufacturing, it may be possible to use computer vision and machine learning techniques on this data to automatically identify key features during manufacturing and to predict the likely presence of defects in manufactured products as a result of melt-pool instability.

As a parallel development in another field, deep learning which is an advanced form of machine learning and based on neural networks, is now sufficiently mature to allow consideration of its deployment in large scale real-world applications including inspection of AM processes. This could result in applying neural networks to automated inspection of the optical datasets collected from an in-situ AM monitoring system.

In this section of this introductory chapter, we will introduce the areas that will be discussing throughout the rest of this thesis, the motivation to work on them and the research questions derived for each. These are then used to frame the research hypothesis and the several research questions which make up the contribution of this thesis, and the experiments that are carried out and reported on later.

1.1 Overview of In-situ Monitoring of Additive Manufacturing

In-situ monitoring facilitates the collection of large amounts of data during the build process in AM. This is illustrated in Figure 1.1 which shows the architecture of the Renishaw InfiniAM Spectral equipped AM system. Each structure in the Fusion module and the InfiniAM Spectral module are numbered and labelled as well as the routes that the input laser and the feedback emissions travel through the system during the manufacturing and monitoring process. Specifically, the monitoring data includes feedback on energy input to the laser and emissions from the AM build process around the melt-pool. It is typically done through two sensor modules: the Fusion optical module that is implemented with the LaserVIEW photodiode and the InfiniAM Spectral module where the MeltVIEW photodiodes are installed.

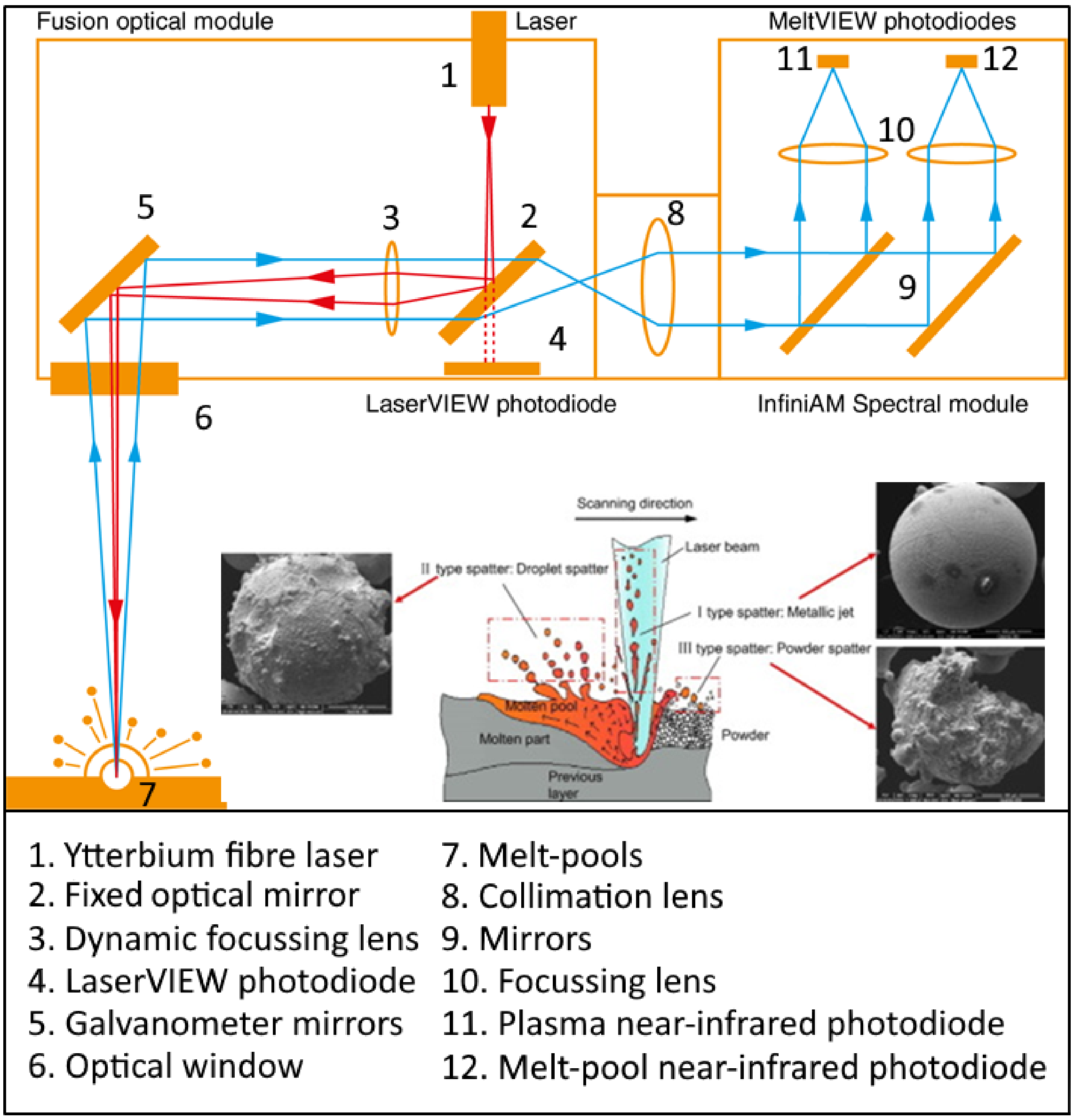

Figure 1.1: The anatomy of an InfiniAM Spectral equipped AM system including input laser, melt-pools, emissions and structures of sensor modules during the manufacturing and monitoring process

In the diagram we can see a laser (1) which is reflected from a 45 degree mirror (2), then focused by a lens (3) before being reflected a second time by another 45 degree mirror (5). Some of the laser will transmit through the first mirror and a photodiode will sense and measure characteristics of the laser (4). When the laser is reflected by the second mirror (5) it hits the melt-pool where it melts the raw material being used and this fuses with the rest of the object or part being manufactured as another layer. Part of the laser is reflected from the melt-pool and back through the mirrors (5) and (2) and the lens (3) and onto another lens (8) where two MeltVIEW photodiodes measure the state of the melt-pool using plasma and near-infrared photodiodes.

The two sensor modules are named LaserVIEW and MeltVIEW and each uses photodiodes as internal sensors for their measurements as follows:

- LaserVIEW: The LaserVIEW sensor captures the energy level in the laser pulse during the manufacturing and measures the intensity of the laser. It is embedded within the Fusion optical module of the Renishaw InfiniAM machine, and captures the filtered light passing through a fixed mirror to gives a relative measurement of laser power to help indicating laser performance during a build.

- MeltVIEW: The MeltVIEW monitors the optical emissions that feedback from the laser melting process during the manufacturing. The two photodiodes of the MeltVIEW are used to measure emissions from the melt-pool. The sensors detect near infra-red plasma emissions in the range 700 nm to 1040 nm and melt-pool emissions in the near-infrared spectrum in the range 1090 nm to 1700 nm.

We now present a more description of the InfiniAM Spectral equipped in-situ monitoring AM system. The monitoring process starts with the tracing of the input Laser beams which are indicated as red lines in Figure 1.1. The laser beams are generated with a power level of 500W from the device at position (1) using ytterbium fibre. Then the laser beams are redirected by a fixed optical mirror at (2) to a group of dynamic focussing lens at (3). During this process, the power levels contained in every laser pulse during a build are detected by the LaserVIEW photodiode located at (4) in order to measure the output intensity of the laser beams. The focused laser beams after passing the lens are further redirected by Galvanometer mirrors (5) so the beams can precisely exit the fusion optical module through the Optical window, which is marked as (6), then focus on the surface of the material being used, thus creating melt-pools (7) and consequently generate optical emissions in the process of the manufacturing. The blue lines in Figure 1.1 are used to indicate the optical emissions, mainly near infra-red plasma emissions and near-infrared melt-pool emissions. The feedback optical emissions travel through the optical window and are then adjusted by the Galvanometer mirrors, the dynamic focussing lens and the optical mirror (marked as (5), (3) and (2) respectively in the figure). After the collection by the Collimation lens positioned at point (8), the emission arrives at the MeltVIEW hardware module. In the MeltVIEW module the emissions are directed again by a set of mirrors (9) to a set of focussing lens (10). Finally, the emissions are detected and measured by the two photodiode sensors of the MeltVIEW module. As mentioned earlier, near-infrared plasma and near-infrared melt-pool emissions are received by the corresponding sensors located at points (11) and (12) as raw data.

For further processing, the DataHUB of the in-situ monitoring system will compile and collate raw data from the LaserVIEW and MeltVIEW sensors in combination with information from the system controller, into a volume that is rendered and stored for viewing and analysis using InfiniAM Spectral software. More specifically, the InfiniAM Spectral software provides 2D and 3D representations of the melt-pool during the AM process for visualisation and further analysis. Further illustrations of these 2D and 3D representations will be included in the next section.

1.2 Introduction to Melt-view Monitoring Data Representation and Research Hypothesis

The data streams collected from the modules which are used in the AM process can be used to build 2D and 3D representations of some of the characteristics or features of the melt-pools for objects being manufactured. Figure 1.2 shows examples of images formed by analysis of the emissions monitoring data from the AM process, shown in 2D and 3D part representations.

Originally, the emissions data collected by the sensors such as those described in Figure 1.1 are formed into 3D point clouds shown as the 3D representation of the emissions features of the objects or parts being built. These 3D representations can

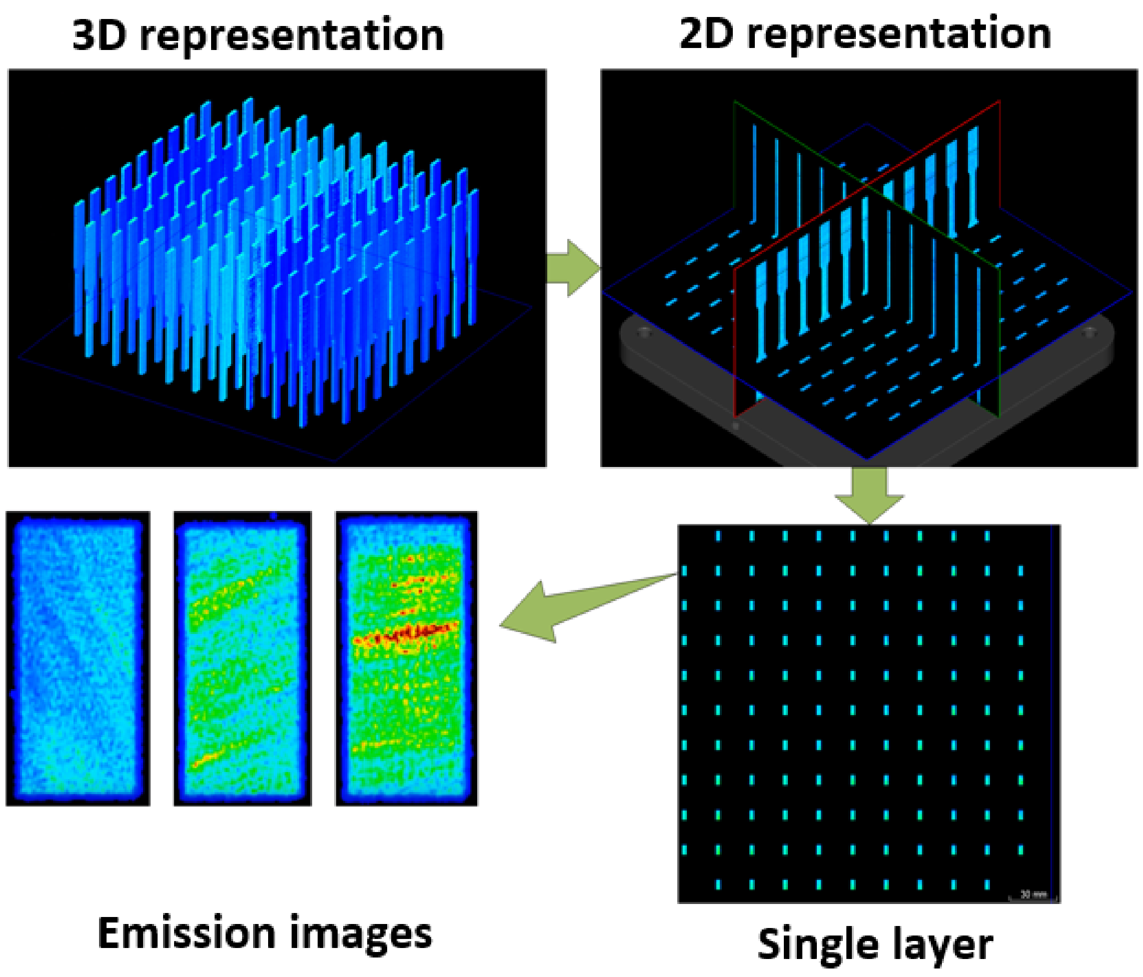

Figure 1.2: 3D and 2D representations of emissions data gathered during the AM manufacture process

be further examined as a series of 2D representations, detailing vertical and/or horizontal cutting surfaces to the level of single layers which have been generated during the printing process. These are in the form of 2D images of melt-pool emissions. By Further zooming in on the 2D images of a single layer, the emissions image of each part for the current layer can be obtained individually and can be used to create an image dataset for further investigation and research into possible defects in the final object, such as the variations in tensile strength mentioned earlier.

Despite the abundance of this type of sensor data from the melt-pool, the analysis and characterisation of emissions data as well as their correlation with defects in the eventual manufactured object, is still a manual process that involves examining the representations produced by the monitoring software. As the performance of human assessors in such tasks can vary, along with fatigue and the associated costs of labor, there is an opportunity for the automation of such inspection. In addition to the cost of labour, defects in manufacturing usually mean a waste of time, energy, and other resources used during the manufacture. Ideally, a real time prediction of a possible or likely defect during the actual manufacturing process itself could tell an operator to either shut down and stop production of the object at an early stage and then to discard or re-cycle the object or to intervene if possible to prevent further defects from happening or even to compensate for defects already present and detected. The end result would be a saving in terms of energy, time, and raw material costs. Achieving this requires not only the ability to better understand and automatically analyse monitoring data, model it, and correlate it with defects, but also the continuous adjustment of such models as new monitoring data is generated.

Deep learning [5], which was briefly mentioned earlier, is a powerful approach to machine learning that applies Neural Network (NN) models to recognise complex patterns and to perform further tasks such as data analysis and automated optical inspection, with high accuracy. However, as the deep learning approach is data-driven, one associated challenge is the demand for properly classified and labelled datasets for the training of the NN model. Unlike many other application areas that have the advantage of large, open datasets for training deep learning models for particular tasks (e.g. ImageNet [6] for image recognition, MNIST [7] for optical character recognition and COCO [8] for object detection), currently there is no large labelled dataset for Additive Manufacturing. In fact, it is expensive and usually not practical in an industry setup to collect such emissions monitoring data and to label it to create datasets for deep learning. It thus follows that better accessibility to labelled data would accelerate the research area of applying deep learning in AM processes. This situation leads us to formulate a research hypothesis for this thesis which can be stated quite simply as follows:

Given recent advances in computer vision and the availability of potentially large amounts of data, or in other words, big data, that are too large and complex to be processed using traditional data processing applications, such as 3D point cloud data with a volume over 1 trillion bytes (1 terabyte), collected from in-situ monitoring of emissions from melt-pools during additive manufacturing, it is possible to use machine learning on this data to automatically identify key features and to predict the presence of defects in manufactured products.

The overall research hypothesis can be broken into three sub-hypotheses as follows:

1. Hypothesis 1 (H1): It is possible to use transfer learning and fine-tuning to create an effective classifier that can perform classification with high accuracy on the emissions monitoring images generated during the AM process for defect detection, even with very little initial training data.

2. Hypothesis 2 (H2): Active learning techniques are a promising approach to automatic labeling of large amounts of unlabeled data collected during the AM process, so that such data can be used for further training and active learning can be used to improve classification accuracy in AM processes.

3. Hypothesis 3 (H3): H3 follows from H1 and H2, and relates to the fact that a systematic combination of transfer learning/fine tuning with active learning can progressively improve the accuracy of deep learning models, specifically for classes of defects where the classification accuracy using conventional approaches is already quite poor.

The research hypotheses which this thesis addresses have also given rise to a number of specific research questions which we address in the next section.

1.3 Research Questions

To evaluate the hypotheses stated earlier, the research in this thesis will focus on the following three specific research questions:

1. RQ1: How can we develop a machine learning model to accurately, (For example, with over 95% classification accuracy) and efficiently classify the problematic and challenging representation of melt-pool monitoring data even with a limited amount of labeled training data? It should be noted that due to variations in data structures, complexity, and applications, there is no strict threshold for defining what constitutes a "limited amount" of data. This criterion needs to be defined according to specific tasks and requirements.

2. RQ2: Starting with a limited amount of labelled data available, how can we create sufficiently large labeled datasets for training deep learning algorithms without spending a large amount of human resources on the labelling task? Once more, due to differences in data structures, complexity, and application, there is no strict threshold for defining what constitutes a "sufficiently large" dataset.

3. RQ3: How can we build and evaluate a framework combining transfer learning and fine tuning with active learning for continuously improving the quality of training data and the resulting accuracy of AM defect detection from monitoring images?

The aim of the research is to investigate how to leverage Deep Learning techniques to support automatic inspection of generated data, specifically for defect detection, as well as the automatic generation and curation of high quality training data, thus reducing manual workload and associated costs.

More specifically, this work will examine two scenarios. For the first scenario, a transfer learning and fine-tuning approach is applied on a pre-trained CNN model to train a deep learning model with only a small labelled dataset. This model would be used to inspect and classify the data collected by the in-situ monitoring suite. For the second scenario, based on the deep learning model developed in the first scenario, Active Learning methods are applied to build a framework in order to automate the data labelling process and to facilitate the generation of more high quality data. Ultimately, the aim is to see this used to help with decisions made by an AM operator during the AM process allowing modification of the output, during the actual manufacture.

1.4 Thesis Structure

This thesis consists of the following nine chapters:

(i) **Introduction**: this is the current chapter, which presents some of the context for the work reported later.

(ii) **Background and Review**: this chapter also presents background information on AM processes with more focus on automatic defect detection. It explores state-of-the-art research in feature extraction from AM process monitoring using machine learning and active learning as represented in the literature. One of the conclusions is that there appears to be very little if any, available datasets to support investigations into automatic defect detection which leads to the systematic search in the next chapter.

(iii) **Systematic Review of Available Additive Manufacturing Datasets**: here we carry out a detailed search of many sources in an attempt to identify datasets in the AM area which we could use for our experiments. Our findings are that there is only one suitable dataset and in order not to overfit our own experiments to just that dataset we will initially use our own dataset for experiments and development, and return to this one outside dataset later.

(iv) **The Design of an Overall System Architecture**: here we introduce the overall architecture of the framework we developed for the research work and it is illustrated in this chapter.

(v) **Defect Detection from AM Emission Images**: this chapter gives an overview of some of the known approaches to improving the performance of machine learning models in order to complete the tasks of feature extraction and classification.

(vi) **Combining Transfer Learning and Active Learning Feedback in Defect Detection**: building on the previous chapters, the methodology of using Active Learning techniques is presented.

(vii) **Evaluation of Our Overall Architecture**: this chapter presents the methods and the results from the experimennts we perform to evaluate the framework introduced earlier.

(viii) **A case study on object detection**: The experiments on our defect classification architecture up to this point are carried out on our own data. This chapter presents an evaluation of our framework for object detection using deep learning on the images of microstructures from AM using an outside dataset.

(ix) **Conclusions**: the final chapter summarises the research presented, revisits the research questions and research hypotheses and discusses whether they have been answered. It also proposes future directions for further research.

Chapter 2

Background and Review

During the AM process, as layers are built up or deposited one on top of the previous in order to make a component or part, the height of each layer and the classification of this height is hugely important, as the next layer on top of this layer will be impacted if the height of previous layers is not within certain tolerances. Thus, an optical or visual inspection task is often used to detect variations in size, orientation of patterns, and other characteristics that could reflect any physical or positional defects in a layer just after it has been deposited. It is also important to develop flexible inspection approaches that can easily be reconfigured for different AM tasks. Many of the conventional approaches to optical or visual inspection of parts are based on feature extraction and further clustering or classification. Over the last decade, advances in deep learning, especially for computer vision applications, have made it possible to apply neural network architectures to automatically inspect the surface of a component and the layers printed during the AM process and to label them using machine learning algorithms. In this chapter, a review of the literature presents work in this area from two aspects: 1. approaches for (visual) classification of the layers in AM and 2. machine learning algorithms for data labeling.

2.1 The Development of Automatic Classification in Additive Manufacturing

2.1.1 Feature-based Approaches to AM Classification

Conventional approaches to AM inspection employ human-engineered feature extraction to find defective patterns in an image. The authors of [9] presented a strategy using thresholds derived from a histogram of gray values for segmentation and further computing the shape describing those features, which were then used in classification. In the work presented later in [10], several types of wavelet models have been introduced to extract features from a surface topography. Subsequently, [11] proposed an approach for defect detection in texture images with an average accuracy of 97% by evaluating the distribution of local gradient magnitudes based on a Weibull fit. There are also other methods for feature extraction based on machine learning in the optical inspection in AM, such as a histogram of oriented gradient (HOG) [12], local binary patterns (LBP) [13], and a gray-level co-occurrence matrix (GLCM) [14].

These approaches represent a broad range of feature extraction methods, and the results generated by these algorithms for feature extraction and defect detection in AM are high in accuracy. However, each of them has limitations in that they often rely on thresholding as part of the classification process. As thresholds are sensitive to variations in background, colors and light, when conditions such as these change, as they sometimes do in AM processes, adjustment of the thresholds will be needed, and that sets back the inspection process. Also, such human-engineered methods are limited by the fact that the features need to be customized to specific AM tasks. In some complex AM conditions, these methods are not robust and discriminative enough to generate results with sufficient accuracy for the associated specific tasks.

2.1.2 Applying CNNs to Inspection During AM Processes

To develop approaches to automatic inspection during the AM process that are both good in adaptability and high in accuracy, several methods based on convolutional neural networks (CNNs) have been proposed. In the work presented in [15] and [16], the authors have applied CNNs in their work and achieved higher classification accuracy than when using conventional machine learning algorithms. However, a major problem with that prior work is that trainung a deep CNN from scratch will require a large amount of data for training. This is one of the characteristics of deep learning, the need for a lot of training data [17]. Thus, one of the challenges in this work is to find a solution that allows the application of deep CNNs when only a limited amount of labelled data is available for training.

In [18], [19] and [20] the authors have used transfer learning to address the problem of little training data availability by using pre-trained weights from a source network to set the weights of a target network and then using the target network to fulfil the task of feature extraction. However, the performance improvements with these approaches to using transfer learning depend on the fact that there is similarity between the source and target domains in their tasks. The authors of [21] have pointed out that if there is a significant difference between the source and target domains, then such transfer learning with fixed transfer weights can yield less accurate results, and this is the case in the work reported to date when applied to the AM inspection process.

2.1.3 Transfer Learning with Fine-tuning

To address the problem of the performance of transfer learning when using dissimilar target and source data, the authors of [22] proposed a method that applies fine-tuning on a VGG-16 network [23]. In their approach, both the source and target networks were based on VGG-16. The only modification on the VGG-16 architecture was reducing the 1,000 node outputs (as used in the source network) to 12 nodes on the target network. For that data, the authors used the ImageNet 2012 dataset as the source dataset and an industrial optical inspection dataset DAGM [24] provided by the German Association for Pattern Recognition, as their target dataset. The overall process of fine-tuning in their approach is illustrated in Figure 2.1.

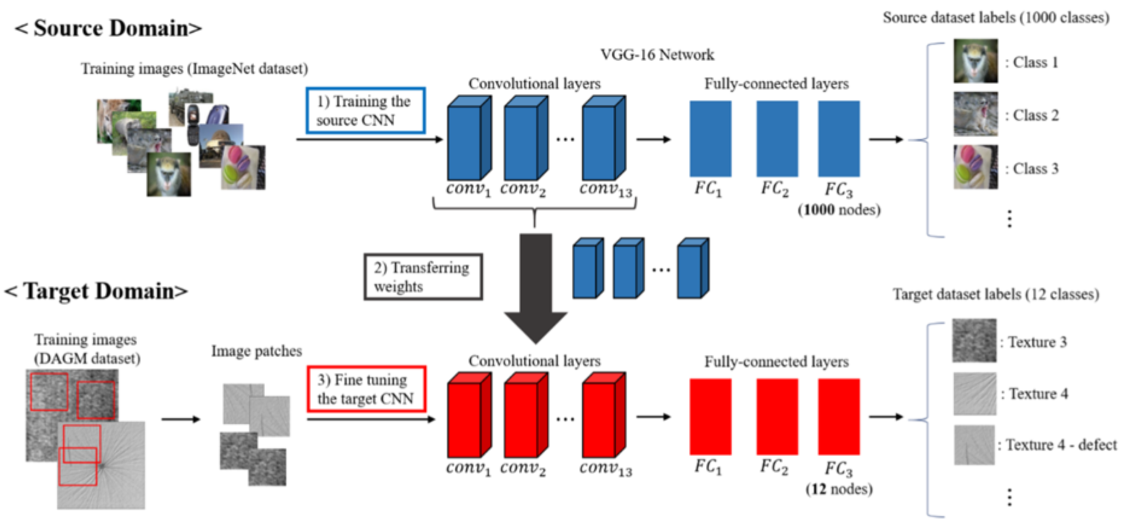

Figure 2.1: Applying fine-tuning on a VGG-16 network from 1,000-class output to 12-class output, taken from [22]

The results show significant improvements on the overall classification performance for all 6 types of AM defects[1] with only 150 positive samples and 1,000 negative samples used in training and achieves 100% on the TNR and almost 100% on the TPR with a lowest value of 99.80 (±0.1) at the third epoch (refers to the one entire passing of training data through the algorithm.) of training, as shown in Table 2.1.

Examining this in more detail, the results show that for certain classes, such as classes 1 and 6 in Table 2.1, training from scratch can result in high performance classification where some of the results approach 100% both in TPR and TNR. The exception to this is for class 4 where classification results are relatively inaccurate as 42.40% (TPR) and 76.00% (TNR) which is not reliable for real world applications. The performance on class 2 is even worse than for class 4 as the TNR is only 1.33% which means after training from scratch using data from class 2, the model was still not be able to verify the true negative samples and "guess" nearly all the uncertain

[1]In this example there are 6 types of AM defects yielding 12 classes, 2 classes (defect / non-defect) for each of the 6 defect types.

Table 2.1: Classification accuracy at the third epoch, results taken from [22].

Texture	Methods		
	Training from scratch	Frozen network	Fine-tuning
TPR (%)			
Class 1	99.73 (±0.2)	91.60 (±1.4)	99.80 (±0.1)
Class 2	95.67 (±2.2)	93.87 (±2.2)	100 (±0.0)
Class 3	85.20 (±9.0)	19.93 (±3.1)	100 (±0.0)
Class 4	42.40 (±17.1)	86.67 (±4.4)	99.93 (±0.1)
Class 5	94.67 (±3.9)	100 (±0.0)	100 (±0.0)
Class 6	98.87 (±0.5)	84.40 (±2.0)	100 (±0.0)
TNR (%)			
Class 1	100 (±0.0)	94.67 (±1.2)	100 (±0.0)
Class 2	1.33 (±1.2)	96.00 (±1.5)	100 (±0.0)
Class 3	99.55 (±0.4)	90.67 (±2.4)	100 (±0.0)
Class 4	76.00 (±10.3)	96.44 (±1.6)	100 (±0.0)
Class 5	99.11 (±0.5)	98.67 (±0.8)	100 (±0.0)
Class 6	100 (±0.0)	98.67 (±0.8)	100 (±0.0)

results as positive samples. After all, when the training dataset is limited in size, it can not ensure the dataset is sufficient for the model to fully understand the pattern from the samples involved.

On the other hand, the general performance of the frozen network that used the transferred weights from the other image domain (in this case, the ImageNet 2012 dataset) is not as high as the good classification results of the training from scratch method. This indeed is a significant improvement on the performance on class 2, though it turns out that the pattern and features of the source domain of the transfer learning for the frozen network can be far from dataset class 3, thus the classification performance falls to 19.93% in TPR, much lower than the value from the previous method. All these problems in the last two methods are solved by further introducing a fine-tuning technique into the data processing, where in the table all the classification accuracy results are quite high values that achieve or are approaching 100%.

2.1.4 Application of Transfer Learning and Fine-tuning in Addititive Manufacturing defect detection

In 2021, the authors of [25] presented their work, which is a machine learning method for defect detection and visualisation in SLS based on convolutional neural networks. In the work, they stated that their best result for defect classification was from a VGG-16 based CNN model and the related dataset which has been used in the classification task in their experiments was openly available to the public. It is worth noting that the data used in this work is from a polymer additive manufacturing machine rather than a metal powder bed machine such as we focus on here, but this is still useful from a data analytics perspective as process prediction and control is required within polymer additive manufacturing as well as within metal additive manufacturing. The work in [25] illustrated a method for non-destructive quality assurance in additive manufacturing processes where they used 2 types of CNN, VGG-16 and Xception networks, and three sets of experiments were reported. These compare the performance of the CNN based models under several experimental setups namely a first experiment with data augmentation, a second experiment without data augmentation and a third experiment with networks that were not previously trained which means transfer learning was not applied to the model. The results of their experiments are shown in Table 2.2.[2]

According to the results presented, performance measures for different machine learning architectures show the effectiveness of defect detection. The experiments proved the pre-trained weights, in the case where they used the weights from ImageNet for the transfer learning process, are important and necessary to successfully train a classifier with a small traning dataset. The authors of [25] also indicated that though Xception is supposed to be a more advanced CNN model than VGG-16, the results in the experiments show that VGG-16 has better performances with the highest value of overall accuracy of 0.971.

[2]This is the first time where the thesis includes confusion matrices in results tables and such confusion matrices will appear in several future Tables and they are denoted by blue background shading.

Table 2.2: Results of the 3 sets of experiments from [25]

Experiment	Model	Confusion matrix	Accuracy	Precision	Recall (TPR)	FPR	F1-Score	ROC -AUC
1st	VGG-16	$\begin{matrix}490 & 10\\ 32 & 468\end{matrix}$	0.958	0.939	0.980	0.064	0.959	0.982
	Xception	$\begin{matrix}459 & 41\\ 65 & 435\end{matrix}$	0.894	0.876	0.918	0.130	0.897	0.934
2nd	**VGG-16**	$\begin{matrix}\mathbf{496} & \mathbf{19}\\ \mathbf{10} & \mathbf{481}\end{matrix}$	**0.971**	**0.963**	**0.980**	**0.038**	**0.972**	**0.993**
	Xception	$\begin{matrix}500 & 0\\ 500 & 0\end{matrix}$	0.500	1.000	0.500	0.500	0.667	0.514
3rd	VGG-16	$\begin{matrix}180 & 320\\ 165 & 335\end{matrix}$	0.515	0.360	0.522	0.489	0.426	0.525
	Xception	$\begin{matrix}500 & 0\\ 500 & 0\end{matrix}$	0.500	1.000	0.500	0.500	0.667	0.526

More importantly, the dataset that was used in the research in [25] is open access to the public. Thus it would be possible for us to use this dataset from AM and carry out further experiments that involve implementing our methods and testing performance. More detail on this will be given in a later chapter of this thesis but for now we will continue with the review of releated work.

2.2 Dataset Labelling

Although applying machine learning in image-based inspection of AM processes can be a powerful approach to high accuracy defect detection, a major challenge is to create sufficiently large labelled datasets for the training process. Manually creating large training datasets is time consuming, expensive, and often infeasible in industrial production settings. Thus, it is important to have an alternative approach to address this problem. For this, two learning methods (i) Active Learning and (ii) Semi-supervised Learning can be considered to allow us to start with a limited amount of labelled training data and enlarge the labeled dataset based on the learning outcome in previous steps.

2.2.1 Active Learning

Active learning [26] is a machine learning technique in which we use smaller amounts of labelled data and a human interactively labels new data points to improve the performance of a model. In recent years Active Learning has been applied in several research areas. For example, the authors of [27] have proposed a novel method called AIFT to naturally integrate Active Learning and transfer learning into a single framework to dramatically reduce the cost of annotation of large datasets of biomedical imaging. Similar approaches were also applied in [28], [29], [30] and [31] in the areas of image database categorisation, segmentation, object recognition and face recognition respectively.

In Active Learning, usually, a small amount of data are labelled and used as the initial training dataset. The rest of the unlabelled data points are referred as the Pool, from which unlabelled data points will be labelled through the Active Learning process. When given an insufficient number of labelled samples in a dataset in which the majority is unlabelled, Active Learning is an algorithm of sample-query-suggestion [26] that can help classifiers to suggest parts of unlabelled samples for active annotation. Initially, it trains a classifier on a small labelled dataset and then the algorithm selects a small number of informative samples based on the current prediction, labels these samples and then enlarges the labelled dataset for further training to improve the accuracy.

In the literature of Active Learning, there are three main scenarios that are considered for the learner to query the existing instances of labels in order to improve the labelling. The three scenarios are referred to as MQS, SBSS and PBS. These 3 scenarios will be described in the next subsection.

2.2.2 Scenarios for Active Learning

In this section, we summarise the major contents associated with each of the three scenarios for Active Learning in which the labels of already existing instances can be queried by the learner. As mentioned previously, these scenarios are Membership

Query Synthesis (MQS), Stream-Based Selective Sampling (SBSS) and Pool-Based Sampling (PBS). In the different scenarios, unlabeled instances are queried to be labeled by the oracle which is normally referred to as the human annotator or simulated annotator in some situations. The three scenarios can be summarised as shown in Figure 2.2.

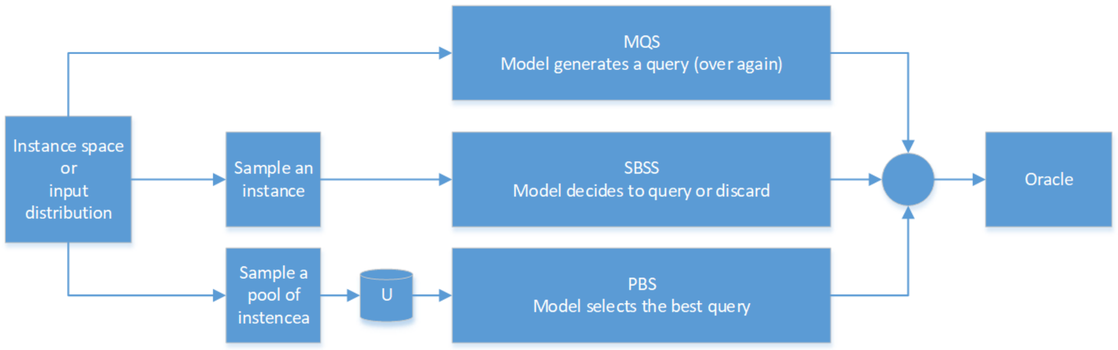
MQS
Model generates a query (over again)
Instance space or input distribution
Sample an instance
SBSS
Model decides to query or discard
Oracle
Sample a pool of instencea
U
PBS
Model selects the best query

Figure 2.2: Three scenarios in Active Learning: MQS, SBSS and PBS

Membership Query Synthesis (MQS):

Membership query synthesis was first introduced early in 1988 in the research work of Angluin [32] where their model was used to generate new queries instead of selecting existing ones. As further explanation, in the Active learning process, the under-training model, also known as the learner, generates instances that are similar to the instances of current learning target following certain underlying natural distributions. The generated instances are then sent to the oracle as a query to label. This Active Learning scenario has been applied to experiments and tasks for processing text and numerical symbols. For example, the authors of [33] have applied MQS in their approach on MNIST and CIFAR-10; [34] have presented a working model for textual Membership Query Synthesis and in [35] the authors have developed a methodology for Semi-supervised sentence classification based on Active Learning via Membership Query Synthesis.

However, the MQS scenario has a limitation in that the queried instances generated by the learner can be difficult or even impossible to interpret for a human annotator. In the research of Lang and Baum [36], they employed MQS with human oracles to train a neural network for the classification of handwritten characters. Many of the query images of characters generated by the learner are not recognisable by the human annotator and cannot be labelled properly. In fact this is also an issue that similarly happens in the application of Generative Adversarial Networks (GANs) [37]. To address this problem, [38] proposed a synthesis strategy that includes a search for the observed nearest neighbours in a feature space, which can synthesise instances close to and spreading along the decision boundary. In recent years, There are also several researches that leverage the advantages of the Variational Autoencoders (VAE) [39] to learn structural information from unlabeled data and use it as an additional criterion in conventional Active Learning to make it more robust against outliers and noise, such as [40] and [35]

Stream-Based Selective Sampling (SBSS):

The concept of Stream-Based Selective Sampling (SBSS) can be tracked back to over three decades ago when the method was first referred as "selective sampling" by the authors of [41]. After that, this scenario was further studied in research from different areas, for instance sensor scheduling [42], learning ranking functions [43] and Neural Information Processing Systems [44]. In recent research, [45] have proposed an approach using blind adaptation in the form of a sliding window to examine the influence of verification latency on the performance of SBSS related Active Learning in order to gain deeper understanding of such effects of the verification latency.

In general, Stream-Based Selective Sampling is based on the assumption that getting an unlabelled instance is free of cost or at worst it is minimal cost. Under this assumption, we then select each unlabelled instance, one at a time. According to the informativeness of the selected instance, the learner will determine whether the label of the instance should be queried. The methods that are used to evaluate the informativeness of samples are the query strategies, which will be further illustrated and discussed in Section 2.2.3.

Pool-Based Sampling (PBS):

Pool-Based Sampling (PBS) was first proposed by [46] in their work as an algorithm for the training of text classifiers. This scenario has since been popularly applied to many real-world research topics and applications of machine or deep learning, and involving Active Learning with classification-related tasks on text [47] [48], images [49] [50] and videos [51] and also extended to regression [52] and ranking tasks [53].

Pool-Based Sampling assumes that there is a large pool of unlabelled data, according to some informativeness measure, also known as query strategies, from which instances are drawn from the pool to be queried. The informativeness measure is applied to all instances in the pool to select the most informative instances of which labels will be requested and this selection activity will continue in iterations until the classifier (or the learner) reaches a certain level of performance, for instance, when overall classification accuracy is over 95% or the budget for queried samples is exhausted.

As further discussion, SBSS and PBS do overcome the major difficulty associated with the MQS method. The main difference between stream-based and pool-based Active Learning is that SBSS scans through the data sequentially and makes query decisions individually, whereas the PBS evaluates and ranks the entire collection before selecting the best query to ask of the dataset and then to present the result of that query for manual annotation. While the pool-based scenario appears to be much more common among application papers, there are settings where the stream-based approach is more appropriate. An example of this is where memory or processing power may be limited, as with mobile and embedded devices. In this thesis the dataset of interest is relatively small in terms of the numbers of samples and the volume of individual samples in this dataset is also small enough to avoid system memory issues. As further consideration, the computational power required for the experiments in our investigations are relatively low as well. Thus, we will focus on the category of Active Learning with pool-based sampling.

2.2.3 Uncertainty Sampling Query Strategies

The main difference between Active Learning and passive learning is the action of the query. Thus it is very important to have proper strategies that can be used to measure the informativeness of the unlabelled instances from the pool in order to create the best query from which to identify further instances to be annotated manually. As stated in [26], uncertainty sampling is the most commonly used query strategy. In this section, three methodologies that are used in uncertainty sampling query strategies are illustrated.

The first method to be introduced is called the least confidence query strategy. Least confidence takes the highest probability for the prediction of each data point, then sorts them from smaller to larger. The formal expression to prioritise using least confidence is defined as:

$$x^*_{LC} = \underset{x}{\mathrm{argmax}} 1 - P_\theta(\widehat{y}|x)$$

where:

$$\widehat{y} = \mathrm{argmax}_y\, P_\theta(y|x)$$

Margin sampling, as the second method, considers the difference between the first and the second highest probability. The data points with the lower margin sampling score would be the ones labelled first. These are the data points the model is least certain about between the most probabe and the next-to-most probable class. Formally, the expression of Margin sampling is defined as:

$$x^*_M = \underset{x}{\mathrm{argmin}} P_\theta(\widehat{y}_1|x) - P_\theta(\widehat{y}_2|x)$$

where:

$$\widehat{y}_1 and \widehat{y}_2$$

are the first and second most probable classes.

Entropy is a concept that originates in thermodynamics and the concept can be re-used to measure the certainty of a model. If a model is highly certain about a class for a given data point, it will probably have a high certainty for a particular class, whereas all the other classes will have low probability. In the case of high entropy it would mean that the model distributes the probabilities for all classes equally as it is not certain at all which class that data point belongs to. It is therefore straightforward to prioritise data points with higher entropy to the ones with lower entropy. Formally, the expression of the entropy score prioritisation is:

$$x_H^* = \underset{x}{\operatorname{argmax}} - \sum_i P_\theta(y_i|x) \log P_\theta(y_i|x)$$

Uncertainty sampling is one of the most popular sampling strategies in the literature [54] due to its simplicity and intuitiveness. It assumes that getting the labels of the least certain samples is more helpful to improving a classifier's performance compared to getting the labels of others. It is considered that the sample having the smallest distance to the decision hyperplane is most uncertain to label and these most uncertain samples are sent to human annotators. To list more examples using the Uncertainty Sampling query strategies, in [55] a confidence value is used as the measure of uncertainty sampling while in [27] entropy is used to evaluate the certainty of classification output for multiple classes. However, in Active Learning there are also alternative query strategies available. The next subsection will further describe some of these alternative approaches that have been used to obtain the most informative instance during the Active Learning process.

2.2.4 Alternative Query Strategies in Active Learning

Although in this thesis we will mainly focus on previous work using the Uncertainty Sampling Query Strategies, it is still worthwhile to have a brief study of additional Query Strategies of Active Learning, which can be potentially useful in the future development of this topic. This also helps us to gain a better understanding of the general methodologies used in Active Learning.

The QBC algorithm was originally proposed by the author of [56]. As it is named, the QBC query strategy involves a committee of the models and all the models of this committee are trained using the current labeled dataset. To have the most informative query, each model in the committee will vote on the label of the current queried instance and then according to the level of disagreement of the votes, the best query will be selected by having the least disagreement among the committee. There are two main approaches to measure the level of disagreement: Vote Entropy [57] and Kullback-Leibler divergence [58].

1. As a further illustration of Vote Entropy, assume a committee with a size C, the range of possible labels y_i and the number of votes that a label receives from the committee $V_(y_i)$. Formally, the expression of Vote Entropy x^*_{VE} is:

$$x^*_{VE} = \underset{x}{\operatorname{argmax}} - \sum_i \frac{V_{(y_i)}}{C} \log \frac{V_{(y_i)}}{C}$$

2. As further illustration of Kullback-Leibler divergence, assume a committee $C = \{\theta^{(1)}....\theta^{(c)}\}$ and $\theta^{(c)}$ is an individual model of the committee and y_i is the true label of the instance. Formally, the expression of Kullback-Leibler divergence x^*_{KL} is:

$$x^*_{KL} = \underset{x}{\operatorname{argmax}} \frac{1}{C} \sum_{c=1}^{C} D(P_{\theta^{(c)}} || P_C))$$

where:

$$D(P_{\theta^{(c)}} || P_C)) = \sum_i P_{\theta^{(c)}}(y_i|x) \log \frac{P_{\theta^{(c)}}(y_i|x)}{P_C(y_i|x)}$$

There more query strategies, such as Expected Model Change [59] that selects instances which induce the largest change in the classifier, Expected Error Reduction

[60] maximising the decrease of loss by adding new data samples, and Variance Reduction [61] minimising the variance of the model in order to obtain the most informative instance. In fact, Active Learning algorithms is a big research topic. To maintain the direction of the study be to relevant to our topic of research, our investigation of Active Learning will pause at this point and the next step will further explore semi-supervised learning, which will be addressed in the next Section.

2.2.5 Semi-Supervised Learning

Compared to Active Learning, semi-supervised learning can also make a classifier learn from both labelled and unlabelled data. Moreover, semi-supervised learning can independently annotate unlabelled samples based on certain rules to improve the performance of the classifier. There are several approaches to semi-supervised learning such as self-training [62], co-training [63] and generative models [64]. The self-training approach is also one of the most simple, efficient, and widely-used approaches and can be used as a wrapper method that applies to existing classifiers. An example of a self-training algorithm is given as pseudo-code as Algorithm 1

Algorithm 1 Example of self-training semi-supervised learning

```
1: Initialize:
2: Given ($X_{train}$, $y_{trian}$) = ($X_l$,$y_l$)
3: while Stopping criteria not met do
4:     Train classifier $C_{int}$ from ($X_{train}$, $y_{trian}$)
5:     Use $C_{int}$ to predict class Label $y_u$ of $X_u$
6:     Select confidence sample ($X_{conf}$,$y_{conf}$); ($X_{conf}$,$y_{conf}$) $\in$ ($X_u$,$y_u$)
7:     Remove selected unlabeled data $X_u \leftarrow X_u$-$X_{conf}$
8:     Combine newly labeled data ($X_{train}$, $y_{trian}$) $\leftarrow$ ($X_l$,$y_l$) $\bigcup$ ($X_{conf}$,$y_{conf}$)
9: end while
```

However, as self-training is based on the assumption that one's own high confidence predictions are correct, a major problem with self-training is that early incorrect labelling by the initial classifier could reinforce the mistakes and this can degrade the performance of a classifier.

To address the problem of self-training following an incorrect line of early misclassifications, [65], [66] and [67] have each tried to avoid such mislabelling of data using post-processing by applying a noise-filtering method. This is based on self-training with editing and nearest neighbour rules respectively but post-processing may edit the class of correctly labelled data into incorrectly labelled data or it may filter out correct data and let incorrect data get into the training process. To further solve the problem, the authors of [68] have proposed two methods, the active labelling method and the co-labelling method.

Active labelling operates as follows: with an Active Learning style, we select the most informative and representative data from among the unknown clusters and then manually give a class label to the selected data. Results show this method can significantly improve the performance of semi-supervised classification. A pseudo-code representation of active labelling is given as Algorithm 2.

Algorithm 2 Active labelling

```
1: procedure ActiveLabelling()
2:     $C_u$ =SemiSupervisedClustering ($X_l$,$X_u$)
3:     for all $c_i \in C_u$ do
4:         Select $x_{cen} \in c_i$ at the centroid of $c_i$
5:         Give class label $y_{new}$ for selected $x_{cen}$ from User
6:         Combine newly labeled data $(X_{train},y_{train}) \leftarrow (X_l,y_l) \bigcup (X_{cen},y_{new})$
7:     end for
8: end procedure
```

Co-labelling is a self-supervised learning method which was developed to overcome the limits of requiring manually labelling as in the active labelling approach by applying an efficient classifier to automatically label selected data in the unknown cluster. The overall performance of this method is not better than active labelling and thus there is a trade-off between automation and accuracy. Pseudo-code for co-labelling is given as Algorithm 3.

In their paper [68] the authors present the performance of the conventional self-training method, the active labelling method and the co-labelling method and compare them using the UCI repository of machine learning, from which 16 datasets were used in their experiments. The results are shown in Table 2.3. The results in Table 2.3 show that, out of the total 16 datasets, active labelling leads to 13 relatively big improvements and 1 relatively small improvement, while the best performances

Algorithm 3 Co-labelling

```
1: procedure CO-LABELLING()
2:     $C_u$ =SemiSupervisedClustering ($X_l$,$X_u$)
3:     Train co-classifier $A_{co}$
4:     for all $c_i \in C_u$ do
5:         Select $x_{cen} \in c_i$ at the centroid of $c_i$
6:         Give class label $y_{new}$ for selected $x_{cen}$ by $A_{co}$
7:         Combine newly labeled data $(X_{train},y_{train}) \leftarrow (X_l,y_l) \bigcup (X_{cen},y_{new})$
8:     end for
9: end procedure
```

(with Random Forest classifier) when co-labelling is used are 7 improvements but 1 degradation. Because of these results, both of these methods should be considered in the work addressed in this thesis. Starting with active labelling could be more effective as it is currently a more stable approach than co-labelling, but co-labelling can also be an option in the later stages of the experiments.

Based on the known advantages of Active Learning and semi-supervised learning, [55] present a clustering-based active semi-supervised classification framework for the inspection of soldering defects, which is illustrated in Figure 2.3. In this work the features of data representation in image format were extracted by a CNN architecture from [23].

The whole of the data set is analysed by k-means clustering algorithm and an initial SVM classifier is trained on the labelled data set. The results of the classifier are then used to evaluate the output from clustering. A simplified homogeneity value decided by the variance of the label distribution is used for the measurement of confidence though the work in [55] only classifies the input data into two classes: qualified and defective. The simplified Homogeneity value is defined as:

$$homogeneity(S_i) = \frac{1}{N_i}|n_p^i - n_n^i|$$

where the dataset X = $\{X_1,X_2,X_3...X_n\}$ is separated into k clusters S = $\{S_1,S_2,S_3...S_n\}$ using k-means, and each sample belongs to the cluster S_* whose centroid C_* is closest to the sample, where N_i is the size of the cluster S_i, and n_p^i and n_n^i are the numbers

Table 2.3: Part of the results from [68] for the comparison of classification accuracy by self-training with originally labeled data, active labelling, and co-labelling with 16 datasets taken from UCI.

Dataset	Accuracy of self-training with ...		
	Original labeled data	Active labelling	Co-labelling (Random forest)
adult	55.24	**59.49**+++	**57.19**+++
bankmarket	68.79	**74.32**+++	**72.11**+++
banknote	88.31	**93.73**+++	88.23
bioNRB	56.05	**60.41**+++	**57.65**++
eeg	71.63	**78.08**+++	71.07
magicgamma	67.14	**69.57**+++	**69.65**+++
mnist17	98.08	**98.71**+++	**98.55**+++
mnist27	95.88	**96.86**+	95.81
mnist38	92.24	**94.27**+++	92.56
mnist49	86.41	**89.95**+++	**88.99**+++
mnist79	91.21	**93.19**+++	90.87
mnist89	95.36	**96.05**+++	95.06
mushroom	96.33	**98.90**+++	96.65
skin	96.93	97.54	96.83
spam	64.07	64.27	64.66
splice	56.45	57.31	53.29–
wilt	67.23	**77.70**+++	**71.89**+++
Increase		14	7
Decrease		0	1

of the predicted positive and negative samples in this cluster respectively. So the homogeneity value is between 0 and 1 for evaluating the confidence of the clusters. Following that, according to the rules of Active Learning and semi-supervised learning, two groups of unlabelled samples are manually and automatically annotated. Then, the initial classifier is retrained on the enlarged labelled dataset and then the process enters the next iteration of labelling and annotation of the samples.

This approach does present a novel classification framework for areas of interest of solder defects based on active and semi-supervised learning concepts and supports human annotators by suggesting parts of the redundant and the informative samples in the labelling process with a relatively small error rate. However, as they did point out, the classifier used is just a binary classifier based on SVM and the feature extractor is directly adopted from the convolutional layers of a pre-trained VGG-16 net. The following points can be improved upon based the work in [55]:

1. The feature extraction architecture of this method was not updated during

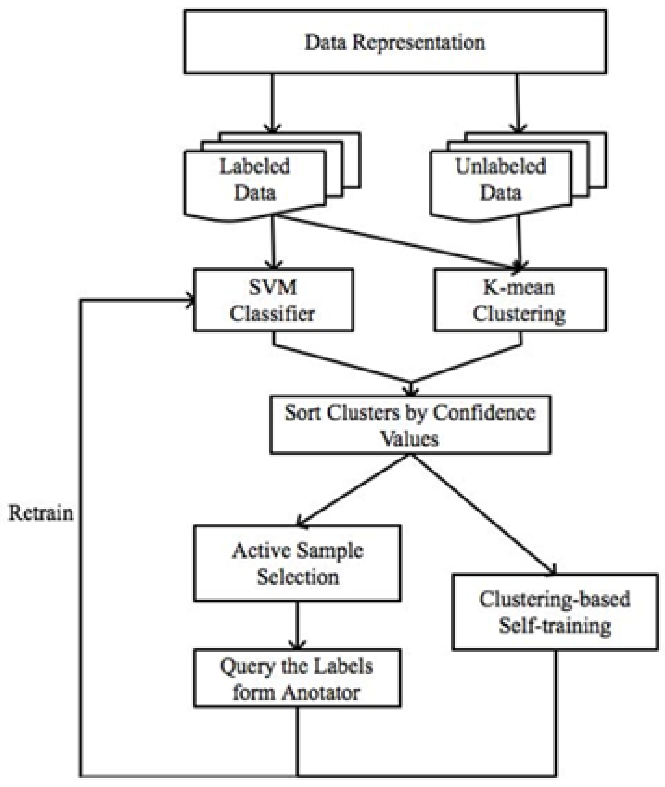

Figure 2.3: A clustering-based active semi-supervised classification framework from [55]

the whole process. As the amount of labelled data grows, this can be used for fine-tuning the feature extractor to potentially give better data representation and accelerate the overall labelling process.

2. At a later stage with sufficient labeled data, the binary classifier can be replaced by other classifiers for better performance and further extending the identification to multiple classes of certain defects rather than only classify the parts as qualified or defective.

This is something we shall incorporate into this thesis later.

2.2.6 Background Literature for Pseudo-Labelling

In the scenario of the "small data challenge" where there are a limited number of labeled examples compared to a vast pool of unlabeled data, Semi-Supervised Learning [69] approaches have proven highly effective. Semi-Supervised Learning leverages both labeled and unlabeled data to improve model performance. Among

Semi-Supervised Learning techniques, Pseudo-Labeling [70], stands out as a simple but highly efficient method in deep learning.

Pseudo-Labeling involves utilising a model trained on labeled data to predict labels for the unlabeled data. These predicted labels are referred to as “Pseudo labels”. It’s important to note that these labels are considered “pseudo” because they may or may not align with the real labels of the samples. In other words, Pseudo-Labeling uses the predictions of the model on unlabeled data as substitutes for actual labels, using the learned representations of the model to assign pseudo labels to the unlabeled samples. This technique effectively enlarges the labeled dataset and introduces semi-supervised learning principles to enhance model performance in the case of limited availability of labeled data. The conventional pseudo-labeling method can be summarised in two distinct stages:

1. Initial Model Training: In the first stage, a model is initially trained using the dataset that contains labeled examples. This initial training phase involves using the available labeled data to build an initial model;

2. Pseudo-Label Generation and Further Training: Once the initial model is trained, it is deployed to generate pseudo labels for the unlabeled dataset. These pseudo labels are essentially predictions made by the model on the unlabeled data points. After generating pseudo labels, both the originally labeled dataset and the dataset augmented with these pseudo labels are combined. This merged dataset, now containing both the original labels and pseudo labels, is used for further training of the model. This additional training phase fine-tunes the model based on the augmented dataset, which includes the original labeled data and the pseudo-labeled data from the unlabeled dataset.

Pseudo-labeling is indeed a powerful Semi-Supervised Learning technique that can improve model performance by leveraging both labeled and unlabeled data. However, it has some drawbacks and limitations as follows:

1. Reliance on Initial Model Quality: Pseudo-labeling assumes that the initial

model trained on the limited labeled data is reasonably accurate. If the initial model is poor or biased, pseudo-labels generated from it may also be inaccurate, leading to a propagation of errors in the semi-supervised learning process;

2. Class imbalance issue: Pseudo-labeling assumes that the distribution of labeled and unlabeled data is similar. A significant distribution mismatch between the two sets may lead to a class imbalance issue in the generated dataset with pseudo-labels;

3. Limited Guidance: If the initial model makes incorrect predictions on the unlabeled data, pseudo-labeling can propagate these errors and amplify noise in the dataset. This can result in a less reliable semi-supervised model. It does not offer explicit feedback or correction mechanisms for the mistakes on labels. This can bring the risk of noise amplification issue when the initial performance of the model is far from optimal.

2.2.7 Background Literature around the Class Imbalance Issue

In a binary classification task with data from two classes, a class imbalance occurs when one class contains significantly fewer samples (minority class) than the other class (majority class).

As has been mentioned above, in many real world applications, abnormal events occurs with relatively lower frequency than normal events, due to the fact that samples of the positive class usually appear and are collected with reduced frequency in contrast to negative samples and this fact naturally leads to skewed data distributions.

Class imbalance issues have be found in various domains, including fraud detection [71], [72], [73], medical science [72], [73], [74], information security [75], [76], and weather forecasting [77].

In the area of machine learning algorithms employed for classification tasks, it is commonly assumed that there is an even distribution of training examples across all classes. Consequently, such models show a bias towards the majority class, using the greater amount of data associated with it. As a result, the trained model tends to show an over-classification of instances within the majority class because of the higher prior probability associated with it, and simultaneously, instances belonging to the minority class tend to misclassification at a greater rate. This phenomenon is particularly problematic when the class of interest corresponds to the minority class, a scenario frequently occurring when conducting classification tasks using machine learning models on AM datasets. In these datasets defects mostly show within the minority subset of the total data population [78]. For this reason, when forming a new set of training data from the results of pseudo-labelling, the class imbalance problem should be considered in order to avoid over emphasis on the major class during the tuning process that updates the classification model.

Generally, to address the class imbalance issue, common approaches for addressing class imbalance can be summarised into 2 main categories, which are also usually referred as having 2 levels in relevant review articles, such as in [79] and in [78].

1. The first category is data level methods that aim to operate on the distribution of the classes in the training set alter dataset in order to reduce the data imbalance effects during the training;

2. The other category is algorithmic level methods that keep the training dataset unchanged but adjust training or inference algorithms where the learning or decision process is adjusted in a way that increases the importance of the positive class. Most commonly, algorithms are modified to take a class penalty or weight into consideration, or the decision threshold is shifted in a way that reduces bias towards the negative class.

One of the common approaches to class imbalance was to use a re-sampling technique, which is a data level method, to make the dataset balanced. This method can be applied either by undersampling or oversampling the dataset [80].

1. Undersampling involves reducing the number of instances in the majority class to match the number of instances in the minority class. The goal is to balance the class distribution by discarding some of the majority class samples. Undersampling can help prevent the model from being biased toward the majority class and can lead to better overall model performance. However, it does mean that the model loses useful information;

2. Oversampling involves increasing the number of instances in the minority class to match or approximate the number of instances in the majority class. This is done by either duplicating the existing minority class samples or generating synthetic samples. Oversampling helps ensure that the model has enough data to learn from the minority class but may also cause an overfitting problem.

Additionally, a more advanced sampling method, SMOTE [81], has been used to address the class imbalance issue. This technique generates synthetic samples for the minority class based on the Self-Organising Map algorithm, ensuring a more balanced distribution between the two classes in the training data. However, it is originally designed for oversampling tabular data. So this approach may not fully capture the complex patterns and structures present in image data, particularly when the images are high-dimensional and contain intricate patterns, textures, and structures. This limitation is especially pertinent in images from additive manufacturing processes, where such complexity is common.

2.3 Literature Survey on Object Detection

We now switch to another topic as part of the background and related work described in this thesis, the techniques for automatically detecting objects in images using computer vision.

2.3.1 Conventional Approaches to Object Detection

Conventional approaches to vision-based object detection employ feature descriptors and feature extraction to find one or many objects in an image. The authors of [82] presented a strategy using SIFT in which a feature is composed of several key points in the image with an orientation and the corresponding descriptor of the area around the selected key points. SIFT key points are searched for through different image scales, known as the DoG pyramid.

A similar method based on machine learning that counts occurrences of gradient orientation in the localised portion of an image is called the HOG [12]. As HOG uses magnitude as well as angle of the gradient to compute its features, its performance is superior than any edge descriptor. These conventional approaches for object detection have been commonly used in many computer vision related tasks such as mitosis detection in biomedical images [83], feature representation and object measurements in microscopy images [84] [85], face recognition [86] and sign recognition [87].

To date, conventional approaches are still effective in various research applications. Yet in spite of that, the rapid developments in deep learning in recent years has raised the performance of techniques which can be used for object detection to an even more powerful level that involves deep learning and R-CNN based architectures as examples of types of deep learning models.

2.3.2 Evolution of R-CNNs

We now illustrate some of the literature marking out progress on the development of R-CNN based architectures. Since 2014 when the R-CNN was first proposed, this type of deep learning model has drawn the attention of many researchers. The model has been further modified and developed with various algorithms to enhance performance and functionalities. Figure 2.4 shows a brief overview of the time steps of the evolution of R-CNN based architectures, which will also be illustrated through the remainder of this section.

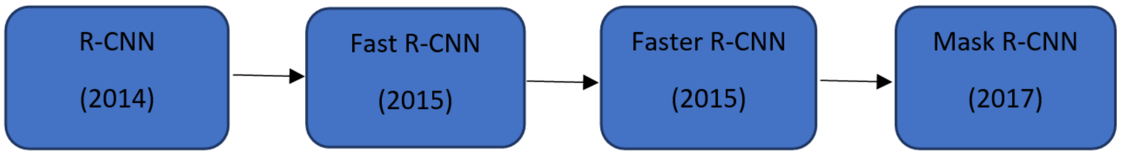

Figure 2.4: Time steps of the evolution of R-CNN based architectures

Region proposals with CNN features

During the period 2010-2012, many conventional visual recognition tasks were based on the SIFT and HOG models, which rely on blockwise orientated histograms. Two years later, [88] proposed research that combines region proposals with CNN features and named it R-CNN. Compared with the conventional methods, R-CNN is an hierarchical, multi-stage processes for computing features that are even more informative for visual recognition than previously. In the research they solved the CNN localisation problem by leveraging the method develop by [89]. That recognition uses regions of the images and has been proved successful both for object detection and semantic segmentation. The architecture of an R-CNN is shown in Figure 2.5 and its work can be summarised as 4 steps:

1. Take an image as input;
2. Extract around 2,000 bottom-up region proposals;
3. Compute features for each proposal using a large convolutional neural network;
4. Classify regions using class-specific linear SVM as well as using liner regression for the related bounding box offset.

However, in its earliest version, R-CNNs also have several drawbacks as follows, each of which need to be addressed:

1. Training a model is a multi-stage pipeline, and this includes the tuning of the CNNs, the training of linear SVMs and the training for bounding box liner regression;
2. The time required for training is long and it also demands space on disk to save the information on features, which in some cases can be as large as hundreds of gigabytes;

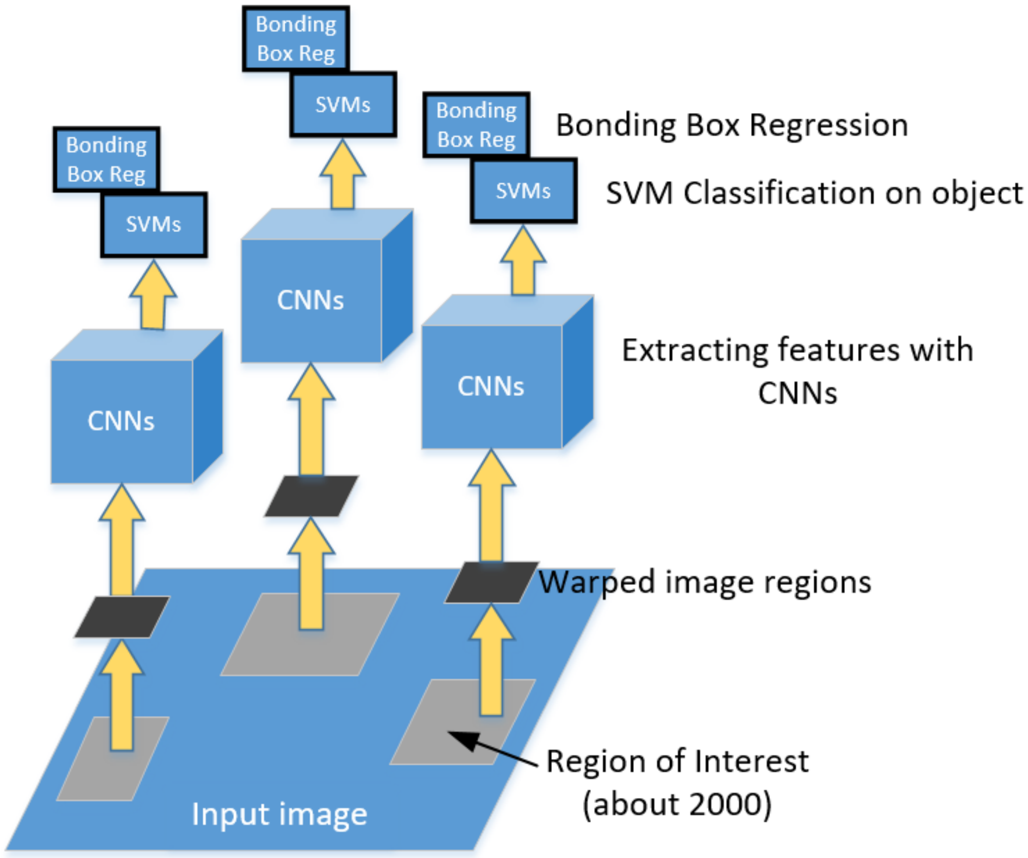

Figure 2.5: The architecture of an R-CNN and its operation in steps illustration from [88]

3. Features are then extracted from each object proposal in the test images. There are up to 2,000 ROI for each of the input images and the computation during the forward pass of the object proposals are not shared among the CNNs. These facts result in comparatively slow object detection.

Fast R-CNN

R-CNN when used for object detection is slow in terms of computation because it performs a forward pass for each object proposal in the convolutional network, without sharing computation. To address this problem, the authors of [90] presented Fast R-CNN which applies further modification based on the original R-CNN. Compared to the original R-CNN, Fast R-CNN employs several innovations to improve training and testing speed while also increasing detection accuracy. In their testing, the model of the Fast R-CNN trains significantly faster than the original R-CNN. More specifically, there are 3 important differences between R-CNN and Fast R-CNN. Firstly, Fast R-CNN uses a single deep convolutional neural network for feature extraction and this speeds up the image processing significantly while R-CNN uses up to 2,000 CNNs for each region of the image. Secondly, Fast R-CNN uses softmax for object classification instead of the SVM classifiers which are used in the original R-CNN for the reason that Softmax has better performance than SVM for the classification in this application. Finally, Fast R-CNN uses multi-task loss in the training of CNNs to increase detection accuracy.

In the architecture of a Fast R-CNN (refer to Figure 2.6), an input image and multiple regions of interest (RoIs) are input into a fully convolutional network. Each RoI is pooled into a fixed-size feature map and then mapped to a feature vector by fully connected layers. The network has two output vectors per RoI which are softmax probabilities and per-class bounding-box regression offsets.

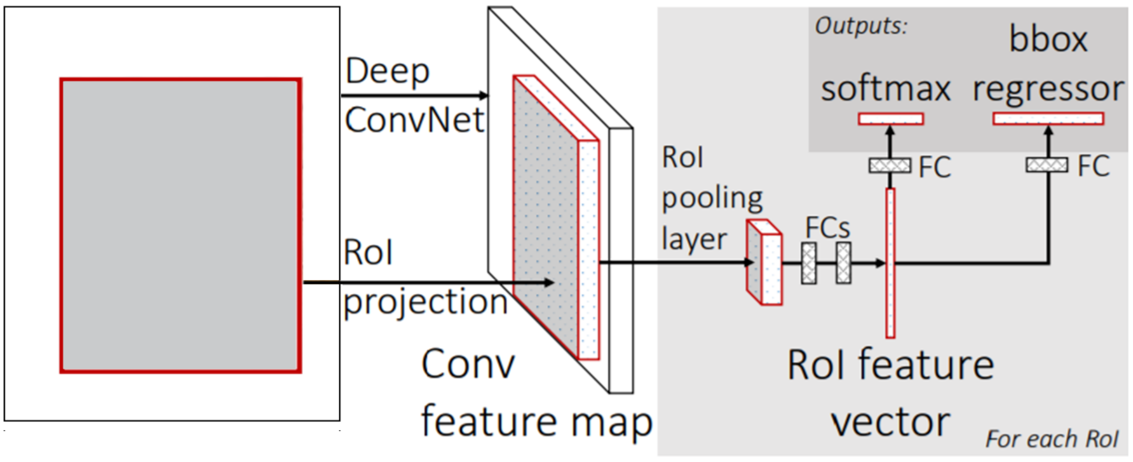

Figure 2.6: The architecture of a Fast R-CNN by [90]

With these improvements, compared to the original R-CNN, the Fast R-CNN has advantages as listed below:

1. Higher detection quality than R-CNN;
2. Training is single-stage, using a multi-task loss;
3. Training can update all network layers;
4. No disk storage is required for feature caching.

At this point we should mention that the RoI pooling layer is a key modification to achieve the usage of a single CNN architecture for all the region proposals in Fast R-CNN. In the RoI pooling layer, max pooling are used to convert the features inside any valid region of interest into a small feature map with a fixed spatial extent with layer hyper-parameters that are independent of any particular RoI. Thus all the input from any RoI can be forwarded to the full connected layer after the RoI pooling layer. However, RoI pooling may cause information loss or misalignment and this issue is addressed later on in the Mask R-CNN technique.

Faster R-CNN

Although Fast R-CNN has significantly reduced the time spent on the object detection process compared to its predecessor, the region proposal step in Fast R-CNN consumes as much runtime as the original R-CNN. The region proposal methods used in Fast R-CNN are still implemented on the CPU and do not effectively exploit GPU power for computing.

To address this problem, Faster R-CNN [91] introduces the RPNs that share convolutional layers with state-of-the-art object detection networks to greatly reduce the marginal cost for computing region proposals. The architecture of Faster R-CNN involves three different neural networks which are the deep convolutional neural network for generating the feature maps from the input image; the Region Proposal Network (RPN) that are used to generate the region proposal that replaces the selective search step in R-CNN and Fast R-CNN; the detection Network which is similar to the Fast R-CNN where it takes input as the feature maps from the convolutional layer and the RPN network in order to generates the bounding boxes and determine the class of the object.

The RPN is the new and key component in the Faster R-CNN. RPN takes an image of any size as input and outputs a set of rectangular object proposals. The input for RPN is the feature map that is generated by the CNN from the input images. Then, features that are further generated from RPN are fed into two sibling fully connected layers one of which is a box-classification layer and the other is a box regression layer for the bounding box to estimate the probability of an object or not object for each proposal and encoding the coordinates of the relevant bonding box respectively. During this process, RPN uses anchor boxes for each sliding window to calculate the relevant scores and coordinates and also helps with translational invariance.

The leveraging of RPN further accelerated the speed of object detection of Faster R-CNN. However, as it still uses RoI pooling, which is the same algorithm used in the previous version (Fast R-CNN), there is possible information lost through the pooling layer. Furthermore, the architecture of Faster R-CNN is capable of generating bounding boxes for objects but it cannot precisely segment each instance of the objects and this is the point where Mask R-CNN takes place.

Mask R-CNN

The Mask R-CNN technique for object detection was proposed by [92] in 2017. It is a new version of an R-CNN-type of deep learning model that can efficiently detect objects in an image while simultaneously generating a high-quality segmentation mask for each instance. Extended from the Faster R-CNN, the Mask R-CNN adds a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. Due to the limit of RoI pooling, Faster R-CNN was not designed for pixel-to-pixel alignment between network inputs and outputs. Mask R-CNN addressed this issue by introducing a new algorithm, namely, RoI Align. With this new feature, the third branch of the Mask R-CNN takes the output from the RoI Align layer and predicts binary class masks. The masks are then used in the task of instance segmentation.

When dealing with floating point numbers in RoI, the RoI pooling algorithm quantises the RoI and this results in inaccurate coordinate prediction. This problem may not be obvious when only a bounding box is required for object detection, but for object segmentation tasks it can be an issue that generates unnecessary offsets because the prediction of an object mask as opposed to a bounding box, requires high accuracy.

To avoid quantisation, the RoI Align algorithm uses bilinear interpolation to compute the exact values of the input features at four regularly sampled locations in each RoI bin, and aggregates the result. The Mask R-CNN was used as the framework by the three best performing teams in the Common Object in Context (COCO) 2017 instance segmentation benchmark, each of which significantly outperformed the previous state of the art.

2.4 ML Applications and the Small Data Challenge in other Industry Domains

The small data challenge in machine learning arises when the available dataset for training a model is relatively limited, containing only a small number of examples or instances. Researchers and practitioners often employ various strategies, such as data augmentation, transfer learning, and regularisation techniques, to address these challenges and build effective models with limited data. For instance, in electronics manufacturing, machine learning is utilised to detect faults in printed circuit boards (PCBs) or semiconductor chips. In a survey conducted by [93], deep learning was highlighted as being widely used in recent years for printed circuit board defect detection due to its excellent performance. However, these approaches often encounter the overfitting problem due to the insufficient training data in real-world applications. To address such issues, [94] employed transfer learning and data augmentation techniques to expand the input dataset and avoid overfitting. Similarly, [95] achieved training with limited samples by utilising a deep siamese semantic segmentation network. Additionally, [96] utilised synthetic data when faced with an inadequate initial training dataset.

The examples provided highlight the challenges faced by electronics manufacturing, in dealing with the limitations of small data within the context of the big data era. Despite the recent successes of deep learning, particularly in computer vision applications, these industries encounter difficulties in developing and deploying machine learning applications due to the scarcity of properly labeled data required for training the models. Solutions such as data augmentation and transfer learning are often employed in the related research to address these challenges.

In additive manufacturing, deep learning applications, particularly those based on computer vision for classification and detection, face similar challenges related to small data availability. Despite the potential for data collection provided by in-situ monitoring, the bottleneck lies in the annotation and labeling of data samples, limiting the application of ML in AM. However, insufficient attention has been given to addressing this challenge, and there has been limited investigation into potential solutions.

2.5 Specific Characteristics of Defect Detection in AM

Defect detection in the additive manufacturing (AM) domain presents several specific characteristics that distinguish it from defect detection in other domains. Some of these characteristics include:

- Complexity of Defects: The defects may manifest differently depending on the AM process parameters, material properties, and geometry of the printed part. this the reason why investigate the patterns at the level of microstructures such as melt-pool and porosity which regardless of the geometry of the printed part.
- Layered Structure: Additive manufacturing builds parts layer by layer, which can pose unique challenges for defect detection. Defects may occur at any layer and propagate throughout the build, requiring comprehensive inspection techniques to detect them across the entire part.
- Material Variability: AM processes use a wide range of materials, each with

its own unique properties and potential defects. Detecting defects in materials like metals, polymers, or ceramics requires specialised inspection methods tailored to the material's characteristics.

- Process Sensitivity: AM processes are sensitive to variations in process parameters, environmental conditions, and machine performance. Small deviations in parameters such as temperature, powder quality, or laser power can lead to defects, emphasising the need for precise monitoring and control systems.
- Data Complexity: The vast amount of data generated during the AM process, including sensor readings, imaging data, and simulation outputs, adds complexity to defect detection. Analysing and interpreting this multi-modal data require advanced data analytics and machine learning approaches tailored to AM applications.
- Non-Destructive Testing (NDT): Traditional NDT techniques used in conventional manufacturing may not be directly applicable to AM due to the unique nature of printed parts. Developing NDT methods specifically tailored to AM materials and processes is essential for accurate defect detection without damaging the part.

In summary, defect detection in the AM domain involves addressing the complexities of layered manufacturing, material variability, process sensitivity, and inherent defects, while leveraging advanced inspection techniques and data analytics to ensure the quality and reliability of printed parts.

2.6 Chapter Summary

In this chapter, several approaches based on feature extraction of optical inspection and labelling algorithms have been outlined. Based on the prior work in the area, it is clear that it is possible to design a framework for an analysis of the data generated from in-situ monitoring of the AM process. Indeed there are other methods associated with semi-supervised learning that could potentially apply to the structures in such a framework and there will be further investigation into these as the thesis develops.

A more detailed illustration of the initial architecture design will be given in the next section. However before we do that, in assessing the literature in automatic defect detection using in-situ data from monitoring of the melt-pool, we have found that there appears to be very little if any, available datasets to support investigations. Thus there is a concern about the public availability of data appropriate for our needs. While we have the capacity to generate and use our own data from our access to AM machinery it is worthwhile carrying out a systematic review of available datasets which is what we do in the next Chapter.

Chapter 3

Systematic Review of Available Additive Manufacturing Datasets

3.1 Introduction to Systematic Reviews

A systematic review of scientific literature is defined as "a review of the evidence on a clearly formulated question that uses systematic and explicit methods to identify, select and critically appraise relevant primary research, and to extract and analyse data from the studies that are included in the review" [97]. The methods used in completing a systematic review must be reproducible and transparent in order that the outputs from the review can be verified independently by others.

While systematic reviews are commonly associated with literature reviews, they can also focus on datasets. However, systematic reviews of datasets differ from systematic reviews of literature in several key aspects, including their focus, methodology, and outcomes.

About the focus, a systematic review of datasets centers on identifying and synthesising existing datasets relevant to specific research questions or topics. The primary goal is to evaluate the quality, completeness, and relevance of available datasets for analysis or research purposes. In contrast, a systematic review of literature aims to identify, evaluate, and synthesise existing research studies and publications related to a specific topic or research question. The focus is on analysing findings, methodologies, and conclusions presented in peer-reviewed articles, books, and other scholarly sources.

The methodology involves systematically searching databases, repositories, and other sources for datasets relevant to the research question. Criteria for inclusion and exclusion of datasets are established, and each dataset is evaluated based on predefined criteria. In contrast, the methodology of a systematic review of literature typically involves searching academic databases, journals, and other sources for relevant research studies and publications. Inclusion and exclusion criteria are set to determine which studies to include in the review. The quality of included studies is assessed using standardised tools, and data from selected studies are synthesised and analysed.

Furthermore, the outcome of a systematic review of datasets may include a list of available datasets relevant to the research question, along with an assessment of the description, limitations, and potential applications of the dataset. This type of review provides researchers with valuable resources for conducting further analyses or investigations, whicle the outcome of a systematic review of literature usually includes a synthesis of findings from existing research studies, highlighting key themes, trends, gaps, and areas for future research.

There are 7 stages of conducting a systematic review according to [98]:

1. Team formation: a systematic review must have a team of two or more; it cannot be completed by one person working alone. In the case of this review, the team consists of the student, with overview of material for the review and outputs from analysis, provided by the supervisors.

2. Questioning: this step usually defines a narrow question which is the subject of the review. [99] describes one common format used to refine the research question into one that is a searchable query which, can be formatted as a question. Since this review is to explore and learn about availability of datasets in AM, as well as to summarise what is available, the question is broader than

usual.

3. Planning methods & strategies: this examines questions such as

    - If the review is feasible
    - Make sure there are no conflicting reviews
    - Planning to carry out each stage of the review
    - Setting goals and timelines for the review
    - Documenting the protocol. Sometimes, in some disciplines this can involve registering the protocol in an open forum such as a journal but this is optional and does not apply.
    - The protocol defines the selection criteria (inclusion and exclusion) for topics to be covered and included in the systematic review.

4. Searching/screening: this is an active step during which the main part of the literature search takes place and it involves the following

    - Searching multiple databases, including the grey literature. This is because systematic reviews should include both published and unpublished literature to avoid a type of publication bias, this is called positive outcomes bias.
    - Once the searching is completed there are then two phases, the first being a screening of the retrieved articles by examining their titles/abstracts (together) and eliminating those that are deemed out of scope. The second phase is a more detailed screening of the full texts of those articles which remain after the first phase and involves confirming that the articles are within scope of the question posed in the review.

5. Managing & reporting: All methods and steps in gathering articles, screening, selecting and summarising, must be fully reported, transparent and reproducible by any third party.

6. Data extraction/synthesising the evidence: Once the relevant articles from the literature are identified and are examined in detail, this phase involves appraising the evidence from those articles, interpreting the results, and performing a qualitative (narrative analysis) and/or a quantitative/meta-analysis.

7. Drawing conclusions, writing and publishing are the final stages of the systematic review.

While some of the above 7 steps described in [98] are not applicable in the case of this thesis, we will follow that general structure of performing a systematic review of image datasets in AM.

3.1.1 Aim of this Systematic Review of Datasets in AM

This systematic review of datasets aims to demonstrate that readily available image datasets from additive manufacturing, which are properly annotated or labeled and openly accessible for training machine learning applications, are insufficient. This insufficiency leads to the small data challenge within the context of the big data era in the field of AM. Further more, the complexity of AM processes leads to variations in the data generated across different applications and laboratory setups. As a result, datasets collected from one AM environment may not be directly applicable to other machine learning tasks or experiments. Even if a labeled dataset is openly available, it may be highly specific to a particular task and less applicable for general training of ML applications. This challenge arises due to the unique characteristics of each AM process, such as variations in materials, printing parameters, and part geometries, which influence the features present in the data. The variability and specificity of datasets collected from different AM environments results in limited availability of labeled data suitable for training machine learning models. This scarcity of diverse and representative data hampers the development and generalisation of ML algorithms for various AM applications. Consequently, researchers face challenges in acquiring sufficient data to train robust models capable of accurately detecting defects, optimising process parameters, or predicting part performance across different AM processes and materials. Thus, addressing the issue of data variability and specificity is crucial for overcoming the small data challenge in AM and advancing the development of ML based solutions for AM.

3.2 Background

Large and open-source datasets of annotated images containing up to millions of training examples such as ImageNet [6], COCO (Common Objects in Context) [8] and Pascal VOC [100] are readily available to machine learning and deep learning researchers. ImageNet for example contains more than 14 million annotated images and the datasets have allowed machine learning to develop hugely over recent years, especially the fact that the datasets are open, easily available and re-used by many researchers. These datasets are usually not specific to any single domain, they are general purpose though there are also many examples of annotated image datasets which are in specific domains. Examples of such domain-specific datasets are in areas like medical imaging with the Cancer Imaging Archive (TCIA) [101], Breast Cancer Digital Repository (BCDR) and Medical Imaging Multimodality Breast Cancer Diagnosis User Interface (MIMBCD-UI) [102] which are images of cancer that have been used extensively. Another example is in the area of recognition of birds where the Caltech-UCSD Birds-200-2011dataset [103] contains images of 200 different bird species.

Given that this thesis addresses defect detection in additive manufacturing, it is reasonable to question the availability of datasets of annotated images taken from the AM process. Acquiring such data is cost-prohibitive in the AM area as shown very recently (in 2022) by Manan and Shao [104]. Several very recent (2020-2022) survey/review papers on the topic of image datasets [105][106][69] each clearly state that labelled samples are often difficult, expensive, or time-consuming to obtain in general. By contrast, unlabelled data can be easily or inexpensively obtained. Consequently, it is desirable to leverage a large number of unlabelled image data for improving the learning performance when this can be combined with a small number of labelled samples. Some researchers have named this as "Small Data Challenges in Big Data Era" [107].

In summary, by carrying out a systematic review we want to see if there is a gap between the currently available image datasets on AM processes and what would normally be available in other areas, namely a large number of standard open-source datasets which can be used in other applications of ML. One reason why this is important is the labelling and annotation work in creating such datasets is a very important but expensive process in the development of additive manufacturing.

As we set out to complete a systematic review on available datasets in AM we have some assumptions which we want to investigate and these form stage 2 of the 7-stage process mentioned earlier, the stage known as "questioning". The assumptions are as follows:

1. A1: There are not many open image datasets from AM processing available for ML/DL training;

2. A2: Existing open datasets from AM are in different online locations and formats, and usually need further processing in order to be used in ML;

3. A3: Manual work for annotation is still required for the majority of datasets when further processing the image datasets;

4. A4: Existing ML applications in AM mainly focus on classification/detection of defects but not in the direction of the "small data change".

These 4 assumptions can also be presented as the following set of specific points to investigate which will help guide the systematic review. To avoid confusion, it worth noting that the points below are regarding the "questioning" stage of this systematic review in this very chapter rather than the main research questions of the thesis.

1. Point 1: Is the current availability of image datasets from AM adequate for computer vision applications? (Corresponding to A1)

2. Point2: For existing open AM image datasets, are they suitable or readily available to machine learning and deep learning researchers? (Corresponding to A2)
3. Point3: Are there many research programs and applications proposed for the development of image datasets that are specified to AM? (Corresponding to A3 and A4)

3.3 Methods

This section defines the selection criteria for the systematic review of the literature. It is an important part of stage 3 of the 7-stage process mentioned earlier, which has been described as "methods and strategies".

3.3.1 Selection Criteria

For inclusion of a dataset in the resulting selection, the datasets should meet the following 5 eligibility criteria:

1. Must be open and accessible;
2. Should be in the format of images datasets (numerical datasets and videos will be excluded and the selection must be valid image datasets and not only images of figures or charts);
3. Must be related to the domain of additive manufacturing, not other 3D-printed work;
4. ML/DL related applications must be involved in using the datasets which may result in different categories;
5. Should be readily available for ML/DL practices i.e. there should be no need for heavy computational processing before putting them into use.

3.3.2 Search Strategy

This section illustrates the search strategy of the "Searching" process, which is part of stage 4 in the 7-stage process. The content of this section involves a description of the search terms and query syntax and the databases where the searching has been carried out.

The databases that have been chosen in the searching stage are DOE Data Explorer, Mendeley, Figshare, Zenodo, AmeriGEO, NIST, Kaggle, DataCite and Google Dataset Search. The names of the involved databases in the searching stage and the corresponding URLs can also be found in Table 3.1

Table 3.1: Websites used in the searching stage

Name	URL
DOE Data Explorer	https://www.osti.gov/dataexplorer/
Mendeley	https://data.mendeley.com/
Figshare	https://figshare.com/
Zenodo	https://zenodo.org/
AmeriGEO	https://data.amerigeoss.org/
NIST	https://data.nist.gov/
Kaggle	https://www.kaggle.com/
DataCite	https://datacite.org/
Google Dataset Search	https://datasetsearch.research.google.com/

Unlike search engines for bibliographic databases which have structures like title, author, abstract, affiliations and other components, most engines for dataset searches do not have these structures. For this reason, the queries that were used in the searches in this section are plain text that contains key words such as: "additive manufacturing" to specify the domain; "image" to define the type of data; "computer vision", "machine learning" and "deep learning" to further indicate the desired applications. Boolean operators such as 'AND' are used to combine the terms to form full queries and filter options have been applied in the relevant databases in order to limit the category and type of results. In some databases, basic searches may not be adequate to retrieve results in a reasonable range and the results usually contain too many non-relevant items. In such situations, advanced searches were conducted to obtain results with better precision.

The initial searches were carried out on May 2th, 2023, and the dates of the search results are up to May 8th, 2023. The search stage is a back and forth process that executes the queries with different combinations of keywords on the selected databases in order to retrieve relevant results.

We now present the results from searching the 9 individual databases and the summary numbers of retrieved results are shown later.

1. From **DOE Data Explorer**:

   - "additive manufacturing" AND image:41
   - additive AND manufacturing AND image: 41
   - additive manufacturing image:41
   - "additive manufacturing" AND image AND "machine learning":3
   - "additive manufacturing" AND "machine learning": 3
   - "additive manufacturing" AND image AND "deep learning": 0
   - "additive manufacturing" AND "deep learning":0

   In summary, the results from DOE Data Explorer are 41 results with 3 of them appearing to be duplicated datasets due to version difference. The resulting 38 datasets may not be in the additive manufacturing domain but this is something we examine in the screening phase of the systematic review process.

2. From **Mendeley**:

   Basic searches: Filter: Data Types: Dataset

   - additive manufacturing image: 40,299 (the range of searching should be further narrowed down to reduce the number of non-relevant results)
   - additive AND manufacturing AND image: 1,008 (range is decreased but still many non-relevant results)
   - "additive manufacturing" AND image: 176
   - "additive manufacturing" AND "machine learning": 83

- “additive manufacturing” AND image AND “machine learning”: 67
- “additive manufacturing” AND image AND “deep learning”: 47
- “additive manufacturing” AND image AND “machine learning” AND “deep learning”: 46

Advanced search on Mendeley Using keywords: any item with keywords “additive manufacturing” AND “machine learning”

- KEYWORDS (“additive manufacturing” AND “machine learning”): 2 (the 2 results are relevant)
- KEYWORDS (“additive manufacturing” AND “deep learning”): 0

Results from basic searches on Mendeley initially yield large numbers of retrieved datasets but mostly highly non-relevant. The scope was narrowed down by applying additional terms such as “machine learning” and “deep learning”, to limit the number of results. However, even doing so there are still a great proportion of the results that are non-relevant, Thus some advanced searches were carried out and 2 relevant results were obtained, but this scope may be too narrow for the current stage.

Summary of results from Mendeley: we take the resulting number of 83 datasets which include the 2 datasets from the advanced search, but this output may contain many non-relevant results.

3. From **Figshare**:

The basic search function on Figshare does not support Boolean operators. For example, if “AND” appears in the searching string it will be treated as a search term. To utilise Boolean operators on Figshare, an advanced search must be conducted. Furthermore, after testing, it shows that the category filter for additive manufacturing on Figshare is case sensitive, so there are 3 terms (additive manufacturing, Additive manufacturing, Additive Manufacturing) to be selected in order to specify the search within the category of additive manufacturing.

(a) Basic search 1:

Category: Deep learning, Item Type:dataset

Searching string: "additive manufacturing": 2

This basic search is to retrieve datasets with the term "additive manufacturing" from the Deep learning category. In fact, the only 2 results are the emission datasets from our research, one is the emission image dataset of positive samples and the other is the dataset of negative samples.

(b) basic search 2: Figshare does not provide a Machine learning category, instead it gives 3 separated categories which are:

Category: Knowledge representation and machine learning, Item Type: dataset

Searching string: additive manufacturing: 39 (results highly non-relevant)

Category: Adversarial machine learning; Item Type: dataset

Searching string: additive manufacturing: 1

Category: machine learning not elsewhere classified; Item Type: dataset

Searching string: additive manufacturing: 6

There are many non-relevant results from the 39+1+6 = 46 results due to the fact that the terms used are additive manufacturing which are regarded as 2 terms as there is no "" surrounding them to make them into a phrase to search on.

We then use a single term "additive manufacturing" under the same filter conditions and obtain the following results.

Category: Knowledge representation and machine learning; Item Type: dataset

Searching string: additive manufacturing: 0

Category: Adversarial machine learning; Item Type: dataset

Searching string: additive manufacturing: 0

Category: machine learning not elsewhere classified; Item Type: dataset
Searching string: additive manufacturing: 2

(c) Basic search 3: filter: Category: additive manufacturing, Additive manufacturing, Additive Manufacturing; Item Type: dataset:

Searching string:

- “deep learning”: 3
- “machine learning”:5

Even if the data type is not limited to images, there are only 8 results obtained.

(d) Basic search 4: Item Type: dataset

Searching string: additive manufacturing image: 270,179

The results from this search are any item with additive or manufacturing or image. The number of results obtained is huge but the contents are not relevant. This result is listed here only to record the search history and for investigation purposes.

(e) Advanced search on Figshare: The aim of the advanced search is to use operator and syntax, such as AND and keyword, to precisely obtain datasets with desired terms and fulfil the task that cannot be done using only basic search. The only filter used in this advanced search is Item Type: dataset

- :keyword:“additive manufacturing” AND :keyword:“machine learning”: 2
- :keyword:“additive manufacturing” AND :keyword:“deep learning”: 3
- :keyword:“additive manufacturing” AND :keyword:“deep learning” OR “machine learning”: 3

3 results have been obtained that have the required keywords.

Summary of results from Figshare: we may take as output, the resulting number of 46 datasets from the basic search 2.

4. From **Zenodo**: Basic search on Zenodo, filter: Typedataset Access Rightopen

   The basic search conducted using the following search strings:

   - additive manufacturing image: 6,625
   - "additive manufacturing" AND image: 10
   - additive AND manufacturing AND image: 4
   - "additive manufacturing" AND "machine learning": 0
   - "additive manufacturing" AND "deep learning": 0
   - "additive manufacturing" AND image AND "machine learning": 0

   The basic search on Zenodo supports search operators and keywords can be directly searched using string and "".

   Summary of results from Zenodo: 10 datasets from the second search can be considered as an initial search output for further screening.

5. From **AmeriGEO**: Filters:None, but when the search is specified to datasets these are the results:

   - "additive manufacturing" AND image: 32
   - additive AND manufacturing AND image: 290 (there are too many non-relevent datasets in this)
   - additive manufacturing image: 6,410 (the range of searching should be further narrowed down to reduce the number of non-relevant results)
   - "additive manufacturing" AND image AND "machine learning": 0
   - "additive manufacturing" AND image AND "deep learning": 0
   - "additive manufacturing" AND image AND "machine learning" OR "deep learning": 0

- “additive manufacturing” AND image AND “machine learning” AND “deep learning”: 0
- “additive manufacturing” AND ”machine learning”: 0

Summary of results from AmeriGEO: 32 datasets can be considered for screening.

6. From **NIST** (with filter: manufacturing and dataset):

   (a) Basic search Filter:Research TopicManufacturing, TypeDataset

   - additive manufacturing image: 77 (the range of searching should be further narrowed down to reduce the number of non-relevant results)
   - “additive manufacturing” image: 35 (reduced the range)
   - “additive manufacturing” image “machine learning”: 0
   - “additive manufacturing” image “machine learning” “deep learning”:0

   Without filter:

   - “additive manufacturing” “machine learning”:0
   - “additive manufacturing” “deep learning”:0

   (b) Advanced search on NIST: As the basic search on NIST does not support Boolean operators. To utilise Boolean operators on NIST, advanced search must be conducted.

   - keyword=“additive manufacturing” AND keyword=“machine learning”: 0
   - keyword=“additive manufacturing” AND keyword=“deep learning”: 0

   The results from NIST are stable in numbers when the terms are additive manufacturing and image, but when further limiting the application to “machine learning” it drops to 0. When using advanced search, it turns out that there is no item that carries the keywords “additive manufacturing” and “machine learning” or “deep learning” at the same time.

Summary of results from NIST: there are 35 results found in the advanced search on NIST.

7. From **Kaggle**: There is no filter option, but searching is specified to a dataset type. The terms on Kaggle date search are treated as tags. It also seems to apply an AND logic between the tags by default. That is the reason why when more terms are added, the number of results are reduced.

    - "additive manufacturing": 6 (the page indicates 7 results, but in fact, only 6 results shown)
    - "additive manufacturing" image: 3 (the page indicates 4 results, but in fact, only 3 results shown)
    - additive AND manufacturing AND image: 3 (same as above)
    - additive manufacturing image: 3 (same as above)
    - "additive manufacturing" "machine learning": 0
    - "additive manufacturing" AND image AND "machine learning": 0
    - "additive manufacturing" AND image AND "machine learning" AND "deep learning": 0

    Overall, on Kaggle the results obtained from the search is much less than Mendeley and Figshare. Even when just a single tag "additive manufacturing" was used, the number of results obtained is still as small as 6 and the queries retrieved the same datasets so there were duplicates and overlaps in the results from individual queries.

    Summary of results from Kaggle: 6 datasets can be considered for screening

8. From **DataCite**: The filter applied in this search is Work Type: Dataset

    - additive AND manufacturing AND image: 43
    - additive manufacturing image: 40
    - additive+manufacturing image: 40

- “additive manufacturing” AND image: 57
- “additive manufacturing”+image: 57
- “additive manufacturing” and image and “machine learning”: 6
- “additive manufacturing” and image and “machine learning” and “deep learning”: 1

Overall, as with previous search systems there are overlapping datasets among the results from different queries.

Summary of results from DataCite: 57 datasets can be considered for screening

9. From **Google Dataset Search**:

- additive and manufacturing and computer vision: 10
- “additive manufacturing” and image:25
- additive and manufacturing and image: 32
- additive manufacturing image: 42

Summary of results from Google Dataset Search: 42 unique and non-duplicated datasets will be input into the next stage

In Table 3.2 we present the summary of outputs from the database searches.

Table 3.2: The numbers of results after searching on the databases

Name	**Number of retrieved results (this is likely to include duplicates)**
DOE Data Explorer	41
Mendeley	83
Figshare	46
Zenodo	10
AmeriGEO	32
NIST	35
Kaggle	6
DataCite	57
Google Dataset Search	42

3.3.3 Screening and Data Extraction

Based on the search results, screening of those results was then carried out. This step is another essential part in stage 4 “Searching/Screening” from the 7-stage process for carrying out a systematic review and during this stage we examined each dataset retrieved from each of the 9 dataset search system outputs, we checked for duplicates across the search results and we examined each result for relevance to the topic of our systematic review and only those which satisfy the criteria presented earlier in Section 3.3.1 are kept. For each dataset the following details from the search results were recorded as the output from the screening and data extraction process:

1. bibliographic information (study title, authors, year of publication)
2. information about datasets generation (Technical background, collection)
3. information about data processing (labelling, annotation)
4. dataset characteristics (size, format)
5. information about the experiment/applications (type of ML/DL involved in the related dataset)
6. impact measures (the number of citations. This may not be an exact value because some of the datasets have no published information/documentation, thus there is no reliable way to know if a dataset has been cited. The aim of including this term is to show when datasets have been effectively used by other research.)

We now present the screening results from each of the databases in turn:

From DOE Data Explorer After screening the 41 results from the search, there are 3 results found from DOE Data Explorer which are all provided by Scime, who is the first author of these datasets. The 3 datasets have similar names but differ in version numbers and the associated DOIs, which are [108], [109] and [110].

Following closer examination, the last 2 datasets [109] and [110] are quite similar, so we may consider to merge them into one which is [110] though currently they are separate DOIs, so we do not treat them as duplicates. Another issue with these datasets is that the access is open but they are not directly accessible. Globus data transfer (https://www.globus.org/data-transfer) is required in order to browse and download these datasets, though this is not against the selection criteria as the access to the dataset is still counted as open.

From Mendeley After screening the 83 results from the search, [108] [109] [110] are duplicates and will not be counted so there are only 4 datasets that meet the criteria, as follows.

Qualified results (1): This dataset [111] is labelled, open for access and a manuscript is provided in the early version.

Possible results (3):

- [112] is not images, but used for a CNN based machine learning application;
- [113] This dataset is small as only 4 images, "The example data should help researchers understand the code and functionality, but you may need to obtain a larger dataset to analyse" – comment by the author of the dataset.
- [114] This dataset is described as large with 1,272,273 images in total, but the original URL link is no longer working, if searched through Apollo, the University of Cambridge Repository. Several datasets with a very similar name can be found and all point to the same author, Douglas. There is more detail on this dataset illustrated in Section 3.4.1

The issue with the results from Mendeley is that there were many non-relevant results among the 83, it also turns out that some of the results are missing authors and some results have been marked as dataset only providing some text work but we are not able to access the actual dataset. For the 3 possible datasets they are highly specific to their original tasks for supporting the purpose and limited approaches to be used, rather than providing general images that can be flexibly used or are sufficient for common ML/DL tasks.

Overall, there are 4 outputs from the screening process.

From Figshare: From the 46 searchresults, 2 datasets are selected, which are [115] and [116] as none of the others pass the screening criteria. We should mention that the dataset in [117] is not properly labelled, thus not qualified at the screening stage.

Overall, from Figshare there are 2 outputs that will be considered.

From Zenodo: Out of the 10 datasets from Zenodo, only one result [118] shows potential to meet our requirements. However, after looking into the related article, it is not an image dataset, but a numerical dataset with some images that are not for ML/DL applications. So there are no suitable outputs that can be obtained from Zenodo.

From AmeriGEO: The 32 datasets from the search output have been screened. However, none of the results are related to ML/DL, none show potential to meet the selection criteria. Thus 0 output from AmeriGEO.

From NIST: The situation with NIST is very similar to that with AmeriGEO. Initially, there were 35 results retrieved as input into the screening stage. Most of the results are related to AM, but none of these datasets are designed for ML/DL applications. Thus 0 output from NIST.

From Kaggle: There are 6 results obtained from searching on Kaggle to be further screened out of which there is just 1 dataset [119] that meets the selection criteria. This dataset is open to the public but it does not have publication information, such as a DOI, as it is not officially published. But this does not go against the selection criteria therefore it is included. Thus, the screening output from Kaggle is 1.

From DataCite: There are 57 datasets can be considered for screening from DataCite and after screening through all the search results, there are 2 datasets that meet the criteria. However these are duplicate results already included from DOE Data Explorer [108] and Mendeley [111]. For this reason, the results will not be counted again to avoid duplication. The final output of new datasets, not already discovered elsewhere, from DataCite is thus 0.

From Google Dataset Search: As Google Dataset Search is a meta-search engine, the 42 results from the searching stage highly overlap with the previous 8 database searches. After applying the selection criteria, there is no non-duplicated result, i.e. each of the results from Google dataset has already been discovered from one of the other search services.

As a summary from the screening process we repeat the entries from the earlier Table 3.2 and add a column indicating the number of datasets after screening and this is shown as Table 3.3. Details of each of the individual datasets are shown in Table 3.4. In the next section we will summarise the usefulness of these 10 datasets for this thesis.

Table 3.3: Numbers of datasets retrieved from the searching and from the screening process

Name	**Number of retrieved results (this is likely to include duplicates)**	**Number of results following dataset screening**
DOE Data Explorer	41	3
Mendeley	83	4
Figshare	46	2
Zenodo	10	0
AmeriGEO	32	0
NIST	35	0
Kaggle	6	1
DataCite	57	0
Google Dataset Search	42	0
	Total (no duplicates included)	**10**

Table 3.4: Summary information on the results from screening

Reference	Technical background	Processing	Size	Format(s)	Type of ML/DL Applications	Number of citations
[108]	layer-wise powder bed images from 20 layers	Labelled, Manual	140 images and files of annotations	.tif .npy	Segmentation	5
[109]	layer-wise powder bed images from two different powder bed printing technologies	Labelled, Manual	6 datasets and files of annotations	.tif .npy	Segmentation	4
[110]	later version of the dataset above	Labelled, Manual	dataset above	.tif .npy	Segmentation	3
[111]	3D point cloud is pre-processed to achieve the target surface of the part. then converted to a 2D depth image	Labelled, Manual	43.4k images	.tif	Classification	no info
[112]	Feature-based CNN network and data for laser powder bed fusion process	Labelled,	no images. Converted to .npy	.npy	Training of CNN	no info
[113]	Using machine learning to predict dimensions and qualify diverse part designs across multiple additive machines and materials	Labelled,	4 images	.tif .csv	Predictions	no info
[114]	Images of the extrusion 3D printing process	Labelled	1,272,273 images	.jpg .csv	Error detection	no info
[115]	emission images from in-situ monitoring of additive manufacturing	Labelled, Manual	150 images	.png	Training, Testing	no info
[116]	emission images from in-situ monitoring of additive manufacturing	Labelled, Manual	150 images	.png	Training, Testing	no info
[119]	**Stainless Steel 316L printed on an ExONE printer**	**Labelled**	336 images	.png	Detection, Segmentation	no info

3.4 Critical Appraisal of Screening Output

From the searching and screening process, it can be seen that there are only a limited number of datasets associated with additive manufacturing which are findable using dataset search engines – 10 in total. Our finding is that even for a dataset which is about the process of AM and which uses ML in some way, the related application is very specific to particular AM tasks or to the manufacture of specific AM outputs that may not be applicable to other processes or outputs in AM. For example, the shapes of manufactured parts in the AM builds can be quite different, thus a dataset for detection and segmentation tasks for building gears, for example, may not be helpful or applicable when the object that needs to be justified changes significantly in shape to a type of gear, for example. In other words, there is a lack of utility across the manufacture of different objects in re-using images. This may also explain the reason why the number of citations to the datasets we have found is low because they are too specific and are not used in other research.

Unfortunately, in the screening results which yielded the pointers to these 10 datasets, the majority of these datasets are used for the detection and segmentation of large components in the manufacture rather than for the detection of microstructures such as melt-pools and porosities which is the focus of this thesis as described earlier in the research questions presented in Section 1.3 of Chapter 1. Detecting defects such microstructures in the melt-pool can be still done even when the geometry of the part changes for example from one gear part to a different gear part. Most of the datasets from the screening results have been created and are available as open access to support some other research question specific to the parts being manufactured rather than focusing on general issues such as porosity and microstructure faults as we do here. In fact, from these 10 screened results only [111] and [110] provided detailed manuscripts which give a description the dataset and describe the value of the development of such a dataset to support microstructure detection in melt-pools.

Apart from the two datasets mentioned above, only one other dataset meets our criteria and that is [116] which is work published from this thesis and which is described later. This shows that there are no openly available databases from the AM process which could be used in ML applications to detect microstructures in melt-pools, apart from the database published from this thesis, and this gives justification for this thesis to create our own dataset in order to apply ML to AM processes and we describe this later in the thesis.

3.4.1 Further Detail on Datasets from the Screening Process

This section delves into each dataset identified during the screening process to provide insights into their characteristics and utility. The discussion on each covers a description, potential applications, accessibility, and any related code or functions, if available. This should be read in conjunction with the results presented in Table 3.4.

1. **Dataset[108]**

- Description: contains layer-wise powder bed images from three powder bed printing technologies: laser powder bed fusion, electron beam powder bed fusion and binder jetting. There are 60 .npy files with manual pixel-wise labelling and 140 top-down layer images from different type of parts, originally used for training and evaluation of deep learning models in [108].
- Potential: has potential to support the advancement of deep learning models for object detection and segmentation however the labelling presents challenges as it requires AM domain expertise to categorise the data. Additionally, classification accuracy remains around or below 50% indicating that the dataset may lack a sufficient sample size.
- Accessibility: openly available but only accessed via Globus, a research cyberinfrastructure developed and operated as a not-for-profit service by the University of Chicago. Information regarding citation and download statistics is not provided.
- Code and Functions: no associated code or functions are available.

2. **Datasets[109] and [110]**

    - Description: [109] and [110] refer to different versions of the same dataset series sourced from laser powder bed fusion (EOS M290 and AddUp FormUp 350 printers) binder jet (ExOne M-Flex printer). Materials include 17-4 PH Stainless Steel, GammaPrint-700, Inconel 718, Maraging Steel, and H13 Steel. Dataset contains top-down layer images from different types of part with corresponding ground truth pixel masks, primarily used for training and evaluation of deep learning models for anomaly and defect detection algorithms.
    - Potential: Similar to the dataset in [108], this dataset facilitates the development and testing of computer vision and machine learning-based anomaly and defect detection algorithms and potentially for object de-

tection and segmentation. Similar to the challenges in [108], the content and arrangement of the dataset are highly specific to the original research, presenting challenges for re-use in other scenarios as well as also requiring AM domain expertise.

- Accessibility: openly available but only accessed via Globus, where download statistics are not provided.
- Code and Functions: no associated code or functions are available.

3. **Dataset[111]**

   - Description: contains images from Fused Deposition Modeling (FDM), also known as Fused Filament Fabrication (FFF) and consists of 434 scan files as heightmap images from the top surface of the 3D printed parts. Each heightmap image is divided into a 10 by 10 grid (100 segments in total) resulting in 43,400 images which are manually labelled and divided into 4 categories: (a) Over Printing Situation, (b) Normally Printed Situation, (c) Under Printing Situation, (d) Empty.
   - Potential: manually labelled and categorised and divided into training, testing and validation sets. Compared to [108] [109] [110], this dataset is more easily re-used in ML applications and can be used for the development of deep learning applications for classification and segmentation in AM.
   - Accessibility: openly accessible on on Mendeley data, providing good accessibility.
   - Code and Functions: the labelling tool (UI) based on MATLAB are available with the dataset but there is no associated function presented.

4. **Dataset[112]**

   - Description: sourced from computer-aided design rather than images captured from an AM monitoring process and not in the format of images.

Dataset is a set of binary files storing numpy arrays (.npy) consisting of features that are combinations of struts with circular and square cross-sections and rectangular walls that are generated by FreeCAD software and a Python script.

- Potential: this dataset focuses on the features of geometries designed using FreeCAD. The dataset with the code for training is open however the format and structure is specific to the original research meaning it can not be conveniently re-used in other common ML tasks without further processing.
- Accessibility: dataset and code openly accessible on on Mendeley data.
- Code and Functions: code for configuration and training of the model in the original paper are available and there is no associated function available.

5. **Dataset[113]**

   - Description: only 4 example images are included to help researchers understand the code and functionality illustrated in the original paper which is about ML models to predict the geometric accuracy and quality of AM parts using feature descriptors such as nominal dimension and feature shape.
   - Potential: the provided 4 images are very limited for ML applications and are not categorised.
   - Accessibility: Dataset and code openly accessible on on Mendeley data.
   - Code and Functions: code for configuration and training of the model presented in the original paper are available. There is no associated function available.

6. **Dataset[114]**

- Description: dataset comprises 1,272,273 labeled images depicting the extrusion 3D printing process. Images primarily capture the printer's nozzle during operation, rather than the manufactured part. Each image is annotated with parameters including flow rate, lateral speed, Z offset, hotend temperature, hotend target temperature, bed temperature, timestamp, and nozzle tip X-Y coordinates. Annotations serve as training data for a multi-head deep residual attention network designed to forecast current printing parameters within the extrusion process.
- Potential: dataset holds significant potential due to its size, however, for broader adoption in machine learning applications several challenges must be addressed. Firstly, the lack of documentation and clear instructions regarding the dataset structure poses a barrier. Secondly, the labeling method and data organisation are closely tied to the original research's approach and model, necessitating additional processing to adapt the dataset for other machine learning applications similar to those encountered in other datasets and highlighting a common challenge in the field.

- Accessibility: Dataset and code openly accessible on on Mendeley data.
- Code and Functions: code to generate results in the original paper are available. There is no associated function available.

7. **Datasets[115] and [116]**

   - Description: these are subsets of an emission images dataset generated from selective laser melting (SLM) developed from 2D representations of emission information generated and collected by an InfiniAM monitoring suite from a Renishaw 3D printer. Images represent the melt-pool conditions of the printed layers in manufacturing parts of Titanium alloy (Ti6Al4V) during the AM process. Images are manually inspected to select 150 images as defected [116] and 150 images as normal samples [115] for use as ground truth.

- Potential: this is a valuable resource for computer vision-based machine learning applications, particularly in classifying melt-pool emissions. Compared to previously listed datasets, this dataset is less complex in structure but smaller in volume though its compactness enhances practical utilisation in image-based computer vision ML applications. This dataset also boasts the highest download count among those listed in Table 3, with over 2800 downloads in total.
- Accessibility: Dataset and code openly accessible on on Figshare data.
- Code and Functions: no associated code or functions are available.

8. **Dataset[119]**

   - Description: this comprises 336 images of Stainless Steel 316L, printed using a Binder Jetting additive manufacturing process on an ExONE printer. Images capture the grain boundaries of the material and are originally intended for training deep learning models to develop universal segmentation methods for grain boundaries. Each of the 336 grain images is paired with a segmentation mask making the dataset readily usable for training machine learning models.
   - Potential: this dataset holds potential for computer vision-based machine learning applications on grain structure segmentation and on applications related to grain size and grain boundary analysis. With a size comparable to the datasets of emission images from [115] and [116], the structure of this dataset facilitates use in machine learning without significant obstacles in terms of domain expertise.
   - Accessibility: Dataset and code openly accessible on kaggle
   - Code and Functions: code used to perform boundary segmentation are available on kaggle. No associated function is available.

3.4.2 Summary of the Screening Results

The screening results which yielded the 10 datasets showed that the majority are used for detection and segmentation of large components in the manufacture rather than for detection of microstructures such as melt-pools with variable porosities which could be used in computer vision-based classification and detection tasks.

This issue is observed across datasets [108][109][110][114] where their specificity to some original research results in difficulties when attempting to apply them to other general machine learning applications in AM. Yet detecting microstructure defects in a melt-pool can still be done even when the geometry of the part changes, for example from one gear part to a different one. Of the 10 screened results only [111] and [110] provide detailed manuscripts which give a description the dataset and describe their value to support microstructure detection in melt-pools, while the dataset that is [116] is of emission images generated from the melt-pools of an AM process. These findings show the limited availability of open and suitable image datasets from the AM domain that meet the requirements for use in machine learning applications, particularly in the context of microstructure detection.

Even though we searched and screened datasets from 9 individual databases, those searches may not have been exhaustive because the 9 databases that we used index only data repositories. There is the possibility of available datasets which are not indexed by the dataset search engines we used and are thus missing from our search. In order to investigate this as a possibility, we searched through the supplementary information in journal articles where research data is sometimes provided by authors. For this we focused on Elsevier's journal "Additive Manufacturing" [120] which is the top-ranked journal in the AM field. In the next section we describe our results when searching the archives of that journal for papers with associated datasets.

Even though we searched and screened datasets from 9 individual databases, that searching for datasets may not have been exhaustive because those 9 databases that we used index data repositories only. There could also be some available datasets which are not indexed by the 9 dataset search engines we used here and thus are missing in the outcome of the searching process. In order to close off this as a possibility, we sought advice from domain experts and, as recommend, we searched through the supplementary information in journal articles and for this we focused on Elsevier's journal "Additive Manufacturing" [120] which is the top-ranked journal in the field . In the next section we describe our results when searching the archives of that journal.

3.5 Additive Manufacturing Datasets as Supplementary Information in Journal Articles

The journal "Additive Manufacturing" is published by Elsevier currently has 72 volumes and the latest issue at the time of writing is 5 June 2023. It is a peer-reviewed journal that provides academia and world-leading industry with high quality research papers and reviews in additive manufacturing. The scope of the journal comprises new technologies, processes, methods, materials, systems, and applications in AM and the journal's impact factor is 11.632 making it Q1.

The total number of articles included in the journal is 3,261 as of May 2023, which is relatively large, and all those articles are searchable on the Scopus website. The searching and screening process, similar to the method used earlier in Section 3.3, was repeated in order to discover articles which satisfy the selection criteria described below.

3.5.1 Selection Criteria for Journal Articles

There are 4 criteria that should be considered in the searching and selecting process in order to be included in the outcome when searching journal papers. Because the journal is already in the domain of additive manufacturing, it is no longer necessary to specify the domain in the criteria. Thus, we can directly focus on the aspect of datasets and machine learning applications. The 4 selection criteria are shown as follows:

1. The articles and related datasets must be open and accessible;
2. Datasets should be in the format of image datasets. Numerical datasets and videos will be excluded and the selection must be of valid image datasets and not only images of figures or charts;
3. ML/DL related applications must be involved in the article for which the dataset is provided;
4. Should be readily available for ML/DL practices i.e. there should be no need for heavy computational processing before putting the dataset into use.

3.5.2 Search Strategy for Journal Articles

The Elsevier journal "Additive Manufacturing" is available on the ScienceDirect website of which the searching engine supports Boolean operators and phrases. The initial searches on ScienceDirect were carried out on May 22nd, 2023, using different combinations of keywords and operators in an iterative manner. When the search process was completed the results were up to date as of May 24th, 2023. To ensure the searching process is reproducible, the filter options as well as the final set of search queries and the associated number of relevant results retrieved per query with no filtering option selected, are shown as follows:

- "machine learning": 204
- "deep learning": 68
- "machine learning" OR "deep learning": 217
- image AND "machine learning": 177
- image AND "deep learning": 64
- image AND ("machine learning" OR "deep learning"): 188

These results indicate many papers published in this journal which address or use some form of machine learning, but we are only interested in those papers which have included a freely available dataset as supplementary information to the paper.

On the ScienceDirect search engine, where searching was conducted, there is a filter option that limits the searching results to be open access and open archives. Although many of the articles are not marked as open, the full text of the articles are still accessible via institution subscriptions, for example, Dublin City University. Thus, we consider such articles still count as open access and we leave the "open access" criteria to be verified in the next step where a further insight will be given on the searching results in order to valid relevant datasets. Searches were then conducted without the "open access" filter option in order to retrieve as many potentially qualified results as possible. Then, the scope was narrowed down by applying additional terms such as "machine learning","deep learning", image and dataset with operators to limit the number of results and retrieve articles that are potentially relevant to the next step, which is screening. The results of this are shown below.

- dataset AND "machine learning": 93
- dataset AND "deep learning": 43
- dataset AND ("machine learning" OR "deep learning"): 96

Summary of results from the Additive Manufacturing journal:
We take the resulting number of 96 articles from the search with no restriction or filter, as the number of articles obtained from search results for further manual screening. This output may contain many non-relevant results but we used this outcome with the aim to not miss possible articles that can further support our research.

For further details, the distribution of the number of the retrieved articles which span the years 2016 to 2023 is shown in Table 3.5. The numbers in the table indicated that prior to the year 2020 there are only 7 research papers related to machine learning or deep learning in the additive manufacturing journal with datasets provided. Since 2020, the number of articles about ML or DL increases each year. Due to the fact that volumes in 2023 are still in progress, there are only 11 results found in 2023 in Table 3.5.

Table 3.5: The numbers of articles in each year from journal searching

Year	**Number of retrieved results**
2023	11
2022	32
2021	25
2020	20
2019	4
2018	3
2016	1
Total	96

3.5.3 Screening of Journal Articles

Screening journal articles for accessibility

All 96 retrieved articles from the Additive Manufacturing journal have been individually manually checked in order to ensure that information on the related datasets can be obtained from those articles. During this process, due to the fact that information on available dataset access can be hidden in certain parts of a paper such as sections that describe methods, results, discussions, conclusions and supplementary documents, each paper was examined until confirmation that the associated dataset can or cannot be accessed. As a consequence, compared to the screening process for datasets in Section 3.3.3 screening all search results from journal articles took longer to complete. After the screening process we found that most articles from the journal focused on illustrating their methods, experiments and results, but only a minority of them clearly indicted how to access the datasets used in their research.

Of the total of 96 papers, the authors of 2 research papers declared that the datasets involved in their work are confidential or they do not have permission to publish the datasets. In another 13 research papers, the authors stated that datasets will only be available on request, which means these datasets are not open to public currently. There are 58 papers that described their datasets or mentioned that datasets were used in their experiments, but they did not indicate the way to access the related datasets. A further 8 articles are review papers with no specific dataset to be screened, and furthermore, there are 4 results are not relevant to the topic of this thesis along with 1 duplicated result that has already been shown earlier in Table 3.4.

At this stage, 86 results have been excluded for reasons of accessibility, non relevance or duplication. The remaining 10 results that provided links to their related datasets are further examined according to the criteria that have been stated in Section 3.5.1, and described in the next sub-section.

Screening journal articles for eligibility

There are 10 journal articles for the final round of screening inspection, during which we will decide if their associated datasets are suitable to be used in this thesis. Table 3.6 shows information on the 10 remaining results in the terms of technical background or topic, the way in which data is labelled (processed), dataset size, format, machine learning applications and the number of citations to the paper. The eligibility of each dataset was verified and analysed in terms of data types and characteristics for the application of machine learning. During the screening process, 9 of the papers and their datasets were further excluded and only 1 dataset remained in the final outcome. The full process of screening is also illustrated in the flow diagram shown in Figure 3.1. In the next section, which is section 3.5.4, we will conduct critical appraisal of the 10 journal articles related datasets to give a further detailed descriptions on these 10 datasets and illustrate the reason for the exclusion and inclusion of each dataset.

Table 3.6: Screening results and related information on the 10 papers with accessible datasets from the journal Addititive Manufacturing.

Reference	Technical background	Data processing	Size	Format	Type of ML/DL Applications	Cited by
[121]	Automated detection of part quality during two-photon lithography via deep learning	manully labeled	over 5GB	tiff, csv, npy	classification	16
[122]	Machine learning for the intelligent analysis of 3D printing conditions using environmental sensor	not image datasets	8MB	csv	classification	8
[123]	MeltpoolNet: Melt-pool characteristic prediction in Metal Additive Manufacturing using machine learning	not image datasets	588kb	csv	classification	8
[124]	Machine learning-based identification of interpretable process-structure linkages in metal additive manufacturing	not image datasets	1.93MB	csv	regression	0
[25]	A machine learning method for defect detection and visualisation in selective laser sintering based on convolutional neural networks	manully labeled	604MB	jpeg	classification	55
[125]	Machine learning and knowledge graph based design rule construction for additive manufacturing	not image datasets	25MB (all files)	x3p	knowledge graph based design	53
[126]	Correlations between thermal history and keyhole porosity in laser powder bed fusion	labeled with parameters	over 10GB	tiff, csv, xlsx	feature extraction, regression	52
[127]	Deep-learned generators of porosity distributions produced during metal Additive Manufacturing	genreated data by GAN	678MB	tiff	Generative Adversarial Networks	1
[128]	Linking process parameters with lack-of-fusion porosity for laser powder bed fusion metal additive manufacturing	not image datasets	16.4KB	csv	regression	0
[129]	Parametric analysis to quantify process input influence on the printed densities of binder jetted alumina ceramics	not image datasets	11KB	xlsx	regression	29

3.5.4 Critical Appraisal of Journal Articles

For a more detailed discussion on the screening results which reduced 10 possible articles with datasets to just 1, during the examination process it was discovered that 6 out of the 10 datasets are not image datasets, such as [122] which uses numerical sensor data that was gathered to record the manufacturing process; [123] and [125] record the (numeric) parameters that describe the characteristic of the melt-pool; datasets descried in [124], [128] and [129] are series of processing parameters of the AM processes. The numbers and terms in these datasets are very specified to the related individual research topic. Because these 6 datasets do not meet the selection criteria that being in the format of images, they have been excluded from the final result.

For the remaining 4 datasets, they contain images and they are open access, but not all of them are ready to be used by other research without further processing. For example, the dataset in [121] is generated from two-photon lithography and

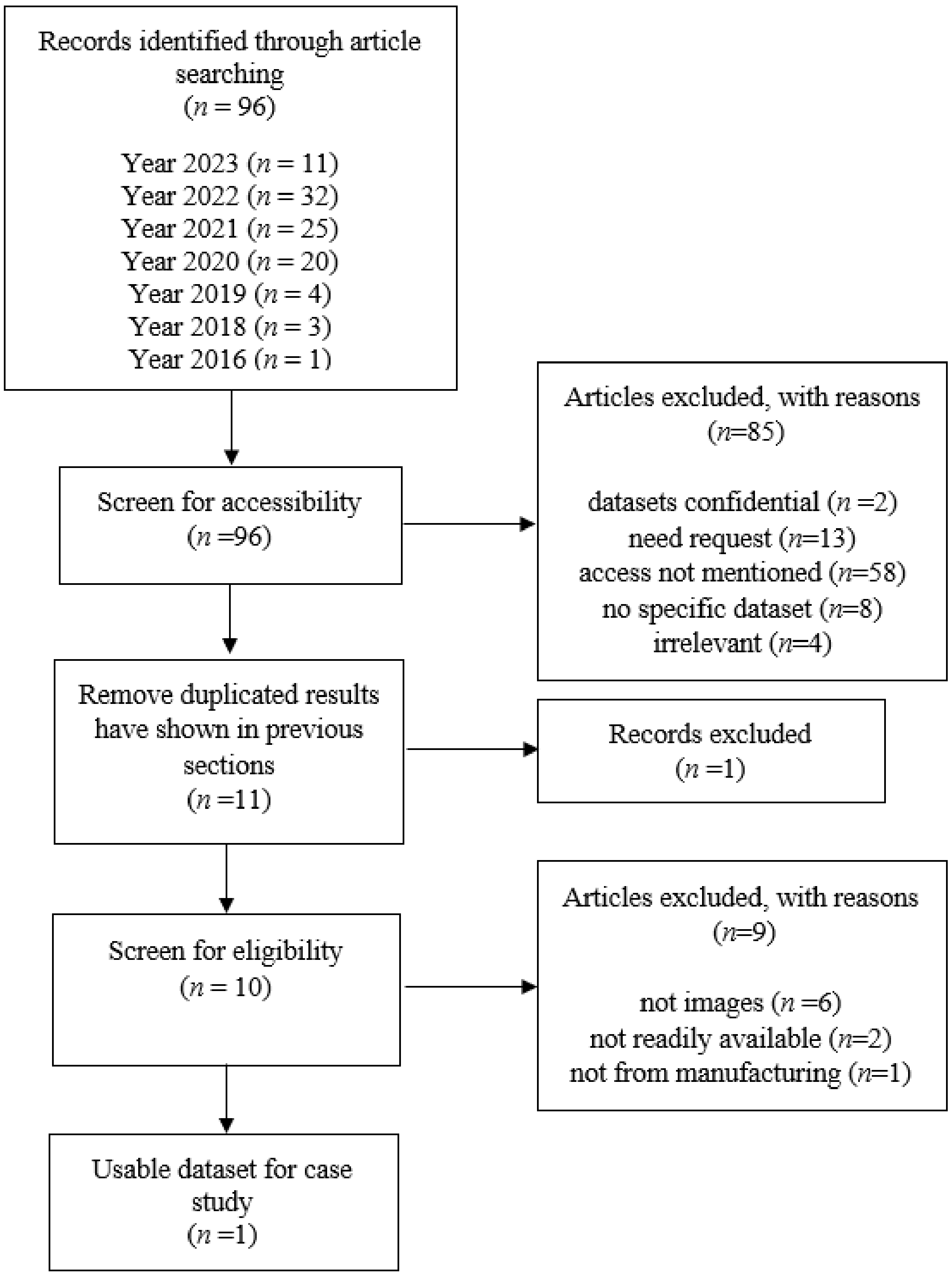
Records identified through article searching
(n = 96)
Year 2023 (n = 11)
Year 2022 (n = 32)
Year 2021 (n = 25)
Year 2020 (n = 20)
Year 2019 (n = 4)
Year 2018 (n = 3)
Year 2016 (n = 1)
Screen for accessibility
(n =96)
Articles excluded, with reasons
(n=85)
datasets confidential (n =2)
need request (n=13)
access not mentioned (n=58)
no specific dataset (n=8)
irrelevant (n=4)
Remove duplicated results have shown in previous sections
(n =11)
Records excluded
(n =1)
Screen for eligibility
(n = 10)
Articles excluded, with reasons
(n=9)
not images (n =6)
not readily available (n=2)
not from manufacturing (n=1)
Usable dataset for case study
(n =1)

Figure 3.1: An overview of the screening process for articles in the Additive Manufacturing journal

the dataset from [126] is based on X-ray and infrared imaging. The methods and experiments in these research papers are novel, but the data structures and the way of labelling are very specific to the associated research. Thus, these datasets are difficult to be used generally and some of the labelling can be hard to achieve or interpret without communication with the experts in the original research team. For this reason, we consider these 2 image datasets are not within the selection criteria of readily available for general practice of machine learning research.

In 1 of the 2 remaining articles, the author of [127] used Generative Adversarial Networks (GAN) [130] to create stochastic realisations of synthetic parts from a limited dataset of experimental parts. This research can be considered as a good attempt to address the "small data challenge". However the dataset that is published corresponding to this research only contains the GAN generated images as their experimental results. The ground truth image data that are used to develop their machine learning model, which we are really interested in, are not included. Furthermore, the GAN generated image datasets only include sample images of pores, resulting in only a single class in the dataset. Due to this fact, the usage of the dataset is very limited for general practice in machine learning. In addition, for better reliability, real datasets from additive manufacturing are preferred rather than datasets generated via deep learning models. For these stated reasons, we consider this GAN generated dataset in [127] is not suitable to be included in our further research in this thesis.

Additional detail of resulting image datasets

To provide additional detail of datasets [121], [25], [126] and [127], they each contain images and are open access, but not all are usable in other research without further processing. For further insight, the descriptions, potential usage, and accessibility of the 4 remaining datasets are illustrated below:

1. **Dataset[121]**

    - Description: contains raw and processed videos capturing three universally encountered polymerisation states of a two-photon lithography (TPL) process. The dataset includes Python scripts which read the video clips and crop out and save regions of interest from every frame.

- Potential: shows potential for benefiting researchers in the fields of additive manufacturing and applied machine learning, particularly in image processing and real-time detection applications. However, the limitations are that the videos lack feature labelling thus requiring human annotators with domain expertise and the TPL process used here is not yet widely adopted on an industrial scale.
- Accessibility: openly accessible on Mendeley data.
- Code and Functions: python script to process video frames and extract RoI as images.

2. **Dataset[25]**

    - Description: comprises images of the powder bed surface of a selective laser sintering (SLS) system, obtained during real manufacturing processes. The 8514 powder bed images monitor and document the quality of the printing process and are labelled into two classes: "OK" and "DEF." The dataset is organised into subgroups, including balanced datasets for training, testing, and validation, enhancing its suitability for common machine learning applications.
    - Potential: primarily used in research aimed at developing machine learning-based methods for non-destructive quality assurance in SLS additive manufacturing processes. This dataset contributes valuable sample images of SLS powder beds. The post-processing and organisation of the dataset facilitates its use by others in applications such as defect classifications and object detection, without requiring high levels of expertise in the AM domain.
    - Accessibility: openly accessible on Mendeley data.
    - Code and Functions: no related code or functions are available.

3. **Dataset[126]**

- Description: consists of simultaneous X-ray and infrared (IR) imaging of laser powder bed fusion of Ti-6Al-4V powder. Fifteen experimental runs were performed at different scan speeds, power levels, and numbers of passes to investigate porosity formation in real time and correlate with the IR signature of the top surface.
- Potential: the dataset comprises raw images, partially labelled with processing parameters from experiments. The total dataset is extensive, and both images and parameters offer valuable information for studying the AM process. However the labelling method and data structures are closely tied to the original research and this requires additional processing to adapt the dataset for other machine learning or deep learning applications. Similar challenges have been observed in previously mentioned datasets [14][15][16] and [20], highlighting a common issue in the field where data may be openly available, but expertise in AM or knowledge from the original research is required before they can be used in ML-based applications.
- Accessibility: The dataset is openly available but can only be accessed via Globus.
- Code and Functions: no related code or functions are available.

4. **Dataset[127]**

- Description: comprises segmented CT scans of 12 Al-10Si-Mg tensile samples produced using Laser Powder Bed Fusion on a SLM 280HL L-PBF printer. The dataset was used for the development of a Generative Adversarial Network (GAN)-based deep learning model and while the CT scan images are not available this dataset contains the GAN-generated images as experimental output.
- Potential: The approach shows promise in generating synthetic parts with porosity, holding the potential to generate more synthetic image data that

can be utilised by other machine learning applications. However, the current output is limited to cylindrical geometry, which may restrict its use by other ML applications in additive manufacturing, as manufactured parts can vary in geometries.

- Accessibility: Code and sample images openly accessible on GitHub
- Code and Functions: code for the GAN based deep learning model is available on GitHub, but this is not within the scope of this review.

Finally, the last article [25] from the searching result proposed a machine learning method based on convolutional neural networks for defect detection in selective laser sintering, which is an additive manufacturing technology that uses a high-power laser to sinter powder of materials into a solid part [131]. The dataset involved in this research paper contains 4,000 images manually divided into 2 different classes for defect detection tasks. The images in this dataset are also clearly separated into 3 categories for training, testing and validation that are ready to be used in the practice of machine learning by other researchers such as in this thesis. In addition, there are total of 8,514 raw images available in the dataset for further potential usage. This paper, which was published in 2021, also has been cited 55 times according to the statistics shown on the Elsevier website, which is the highest number of citations among all papers in Table 3.6. As this dataset of defect detection in AM shows very good potential that can be further unitised, we consider it can be used in this thesis as part of a case study.

In summary, originally we targeted datasets which are images of microstructures, such as in the melt-pool, from a monitored AM process. However, after scanning through all relevant articles in the journal, we find there are no image datasets of the melt-pool that are open access and readily available as supplementary information from the journal. During the searching and screening process we discovered some papers, such as [123] and [125] that described datasets related to the melt-pool, as well as [128] that mentioned research related to lack-of-fusion porosity. However, these datasets are in numerical format which record processing parameters and characteristics of melt-pools during the AM process. These datasets may be good support in some other machine learning or deep learning applications but are not suitable for computer vision based ML/DL approaches and tasks.

Thus, we can not utilise these datasets directly in our computer vision based research and experiments. Furthermore, out of the 96 results from the selection stage, there are 58 papers that did not provide information on access to their related datasets. This fact could be considered as evidence proving that not enough attention has been paid to providing open datasets in the AM domain.

3.6 Chapter summary

This systematic review of available datasets for image-based defect detection in additive manufacturing processes comprised two primary investigations. The first investigation covered a comprehensive search across nine dataset databases, while the second focused on the highly-regarded Additive Manufacturing journal. We have conducted thorough searches through the dataset databases and the journal archives, followed by manual examination and screenings of identified datasets, according to a set of selection criteria. The review reveals a significant gap in the availability of open and readily available image datasets in the field of defect detection in additive manufacturing.

Our conclusion is that it is essential to have appropriately annotated datasets for the effective application of machine learning, particularly computer vision-based deep learning in the context of additive manufacturing. Without such datasets we will not see improvements in automatic defect detection in AM processes and the potential that this brings in terms of improved quality control and savings during manufacture when defects are detected and can be rectified or manufacturing of the remaining deposition layers prevented, will not be realised. The search results indicate a scarcity of publicly available datasets that meet the criteria of being open, accessible, in the appropriate format, relevant to AM defects, and robustly adaptable to various ML applications.

After extensive and structured searching of both dataset repositories and journal article supplementary information for a high-impact journal in the field we found just one suitable dataset, described in [25]. This result supports our preliminary conclusion from Chapter 2 that there are few datasets for applying machine learning to defect detection using in-situ monitoring of the melt-pool which are open and available. While the dataset in [25] is from a polymer additive manufacturing machine rather than a metal powder bed machine, this is still good from a data analytics perspective as process prediction and control is required within polymer as well as in metal additive manufacturing. The experimental method used to collect the data in [25] is reasonably well defined however to repeat their experiments precisely the authors would also have to have detailed the geometry of the part being manufactured, the hatch spacing, laser scan pattern, and the power setting used during production. While the paper gives total power of the machine it does not give the power that was used during part production and image capturing. This absence of sufficient data to repeat the experiment is not uncommon in the area even for papers in the Additive Manufacturing journal which is as high impact as it gets in manufacturing journals. The absence of this additional data would be a problem if the research was purely focused on the effect of manufacturing process parameters but it is not an issue in this thesis as we focus on defect detection, not the effects of such defects on the manufactured part.

In summary we believe it is OK for us to use this dataset in this thesis because there is a stochastic nature to the additive manufacturing process and our focus is on data analytics of the dataset which has already been classified into non-defective / defective subcategories. We intend to use this discovered dataset in experiments where we use it as a case study later in the thesis and our first application is to use data that we gather outselves.

In the next chapter we will describe the architecture of our proposed system.

Chapter 4

The Design of an Overall System Architecture

This chapter will introduce and present the conceptual design of the overall architecture for the proposed framework to address the research questions listed in Section 1.3 and contribute to validating the hypothesis presented in Section 3.2.

Although the overall architecture is designed to work as a whole framework, the design process has been carried out step-by-step. At each step, we reviewed and identified state-of-the-art approaches for addressing a specific challenge, then according to the outcome of such analysis and the availability of necessary materials and facilities, we finalised the associated design, implementation and test as well as further adjustment of the overall development of the architecture. Ultimately, the conceptual design was the result of an iterative process aimed at the validation of our hypothesis. Mainly, we would like to divide this conceptual analysis work into three steps as listed below:

1. In the first step, we focus on the problem caused by the limited size of the dataset. The target is to design an approach that leverages the advantages of deep learning models rather than conventional machine learning models in order to develop a classifier which is not only efficient with limited available resources including data and computational power, but also to generate a

solution that can be easily adapted to different types of image patterns.

2. In the second step, we further improve our approach but in another direction. We still target the development of the classification model, but we build upon the outcome of the first step to improve the effectiveness of the training process using active learning techniques. Improvements in performance are determined by using active selection of samples from the dataset and continuously fine-tuning the model by performing active query and learning iterations.

3. In the final step, we focus on how the conceptual design can support automated labelling and we run validation using different available datasets and collect performances metrics for evaluations.

4.1 Step One: Deep Learning Model with Transfer Learning and Fine-tuning

To design a defect detection and automated labelling framework, the first challenge is to address that our research is limited by the available data samples for the basic training process as mentioned as the first research question in section 1.3. Indeed, a substantial amount of work had to be devoted to accessing the raw data and preparing them to be used, with multiple challenges.

The raw data used in this study were collected from the InfiniAM Spectral software, specifically designed to operate in conjunction with Renishaw's additive manufacturing process monitoring system. The raw data are sensor data collected and stored in the format of 3D point clouds that can only be accessed and visualised through the InfiniAM Spectral software. Thus, they need to be further processed before being utilised in any applications. A detailed description of this monitoring system can be found in Section 1.1, as depicted in Figure 1.1. The specific Renishaw machine employed is the RenAM 500Q. Detailed operational information about this machine can be accessed online at `https://www.renishaw.com/` The material used in the manufacturing process is a titanium alloy powder, Ti6Al4V. The printed parts consisted of 117 dog bone-shaped testing pieces, although the recorded data primarily pertains to the cutting surface printing layer wised. Consequently, all the testing pieces appear as rectangles with dimensions of length (L) measuring 1 cm and width (W) measuring 0.5 cm. Due to the limitation of accessibility, comprehensive details regarding the dimensions of the dog bone-shaped parts are not accessible. The process for developing the emission image dataset from the video record will be described in the following paragraphs.

At the time of our initial access to the data source, although it was possible to observe the 3D and 2D representations of the emission data via the monitoring window, the spectral software which is the data processing and visualisation tool associated with the in-situ monitoring system of the InfiniAM machine had not been developed to have functionality to properly export the captured data into images in a standard format. In order to export the emission images with a semi-automatic method, we used a clicker macro to advance the monitored layers one by one in a selected sequence in the monitoring window with a short time delay between each clicking. Meanwhile, we also activated a screen capture software application to record all the emission images continuously appearing in the window of the spectral software.

As a result, we had to go through a relatively large number of emission images of the 2D representations originally stored internally in the Spectral software and then transform the relevant emission images from the 2D representations in the monitoring window into a video. The video had to be further processed and separated into single frames and duplicated frames had to be removed to only keep a single image for each AM layer. Only at this point were the images are finally exported into a set of Portable Network Graphics (PNG) files with a resolution of 440×840 pixels. These images still need to be manually labelled to create the initial training dataset for the development of the classifier (see Figure 4.1).

The created dataset is highly unbalanced due to the fact that positive (or abnormal) samples are a minority. This means that the initially created dataset becomes relatively small in size if we want to restore the balance between positive and negative samples.

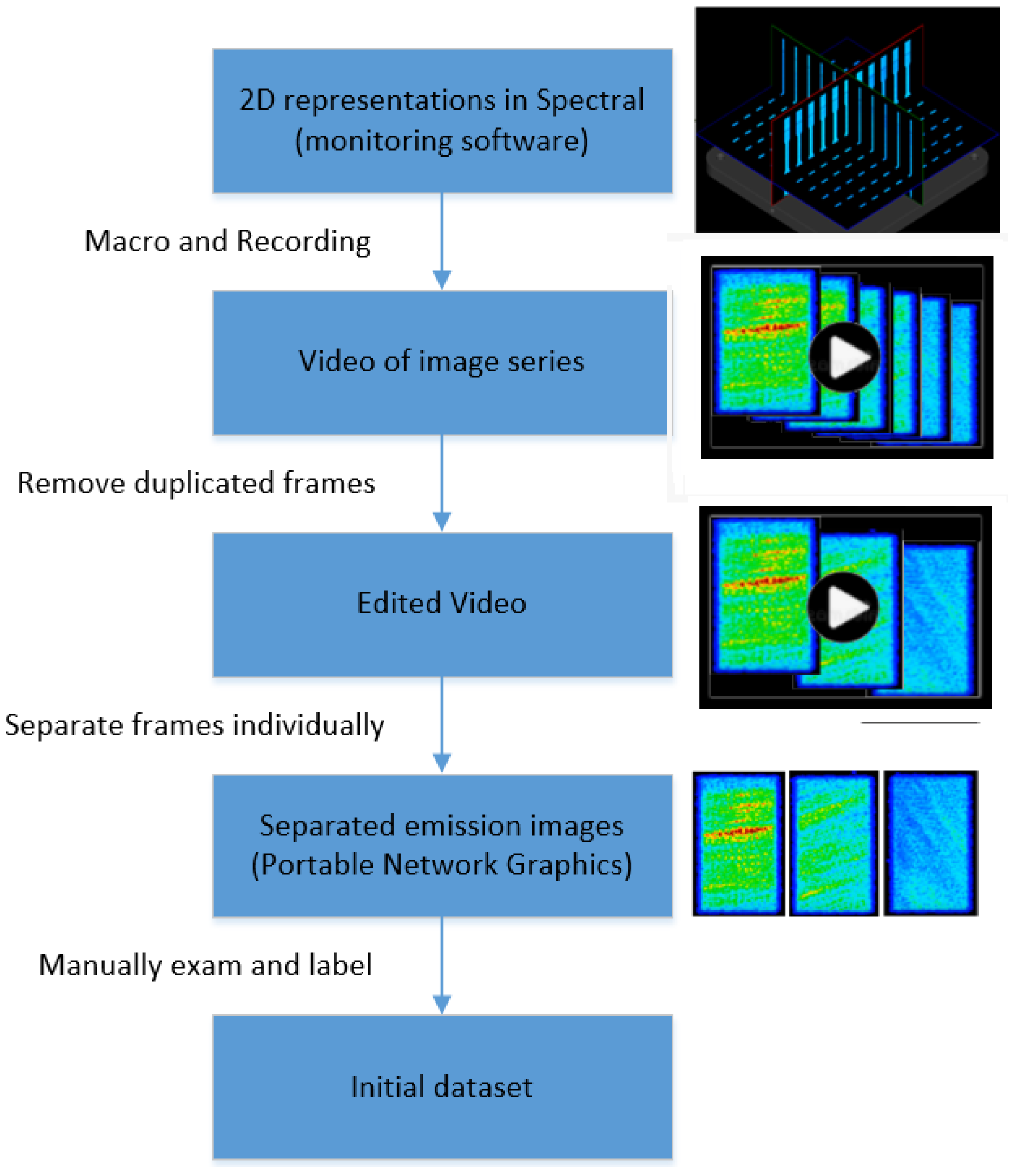
2D representations in Spectral (monitoring software)
Macro and Recording
Video of image series
Remove duplicated frames
Edited Video
Separate frames individually
Separated emission images (Portable Network Graphics)
Manually exam and label
Initial dataset

Figure 4.1: Steps to export emission images from the original data source

The work reported earlier in Section 2.1.3 showed the effectiveness of an approach that applies transfer learning with fine-tuning on a VGG-16 deep architecture for classification when training data is very limited. This is the point where our basic model design and development begins. The initial classifier in the overall architecture of the framework is designed based on the convolutional layers of a VGG-16 model with transfer learning for feature detection and extraction, modified with custom neural layers for the classification task and output results according to the extracted features. The approach will be presented in detail in Chapter 5, including the creation of the initial datasets, the comparison of the performances between different setups and modifications of the classification model as well as the research for the implementation of the CNN based initial classifier. It will also include the experiments on the emission image dataset along with other different industrial patterns in order to investigate the relevant performances and to apply relevant modifications to improve the model.

In the early stages of the architecture design in this research, we have also considered another type of deep learning model, namely autoencoders [132]. However, there are several problems associated with autoencoder based approaches.

Autoencoders are mainly designed to encode their input into a compressed and meaningful representation, and to then decode it back such that the reconstructed input is similar to the original one. In this way the Autoencoder approach generates new samples and is able to augment the dataset. However, the generated samples may not follow the real patterns that can be formed during the manufacturing process. In addition, there can be situations where the generated samples are difficult to judge and annotate even for human experts. This can bring too much noise and uncertainty into the output and can make it difficult to successfully train the model. Furthermore, the process of encoding and decoding for the generation of new samples would consume additional computational power and possibly demand a larger amount of resources and increased training time.

For all the above reasons, we decided to focus on transfer learning and fine-tuning to increase the number of labelled data based on the existing real samples from the additive manufacturing process rather than increasing the size of the dataset by adding generated samples. Should it be necessary to increase the total number of training samples, we would consider data augmentation rather than generating samples via approaches based on autoencoders.

4.2 Step Two: Improving Training with Active Learning

Once the concept of an initial classifier is finalised, we tackle research question 2 from Section 1.3, and we aim to discover methods and algorithms to increase the number of properly labelled samples in the dataset but without spending a large amount of human resources on the labelling task. As discussed in the literature review, active learning techniques can be a powerful approach to accomplish such a kind of task. We estimate that by combining active learning with fine-tuning techniques, we can further develop our framework into a human-in-the-loop mechanism that involves a human annotator in the training process of the classification model (referred to as the "learner"), in order to further improve the efficiency of data usage in the overall training process and obtain classification results with a higher level of accuracy.

Figure 4.2 shows the data flow in the conceptual design including the active learning process. In this development stage, the trained classifier is used for predicting labels and evaluating the confidence via query strategies in the active learning route. This is also the stage where the most informative samples are selected as high priority and queried for annotation. In fact, at this stage, the approach already begins to help with labeling in an active manner by indicating which samples can be more effective for the training of models if labeled and additionally included in training. This in turns accelerates the creation and growth of a high quality training dataset speeding up convergence in the training process.

This is an important step to support the functionality of this framework which enables automatic labelling. As the samples classified by the deep learning model will be added to the labelled dataset, the newly obtained samples are also involved in the next cycle of fine-tuning or retraining. A classification outcome with low accuracy may result in labelling failure that possibly reinforces itself through subsequent similar errors in the subsequent labelling tasks. Details of this particular component of the framework, including the implementation and setup are described later in

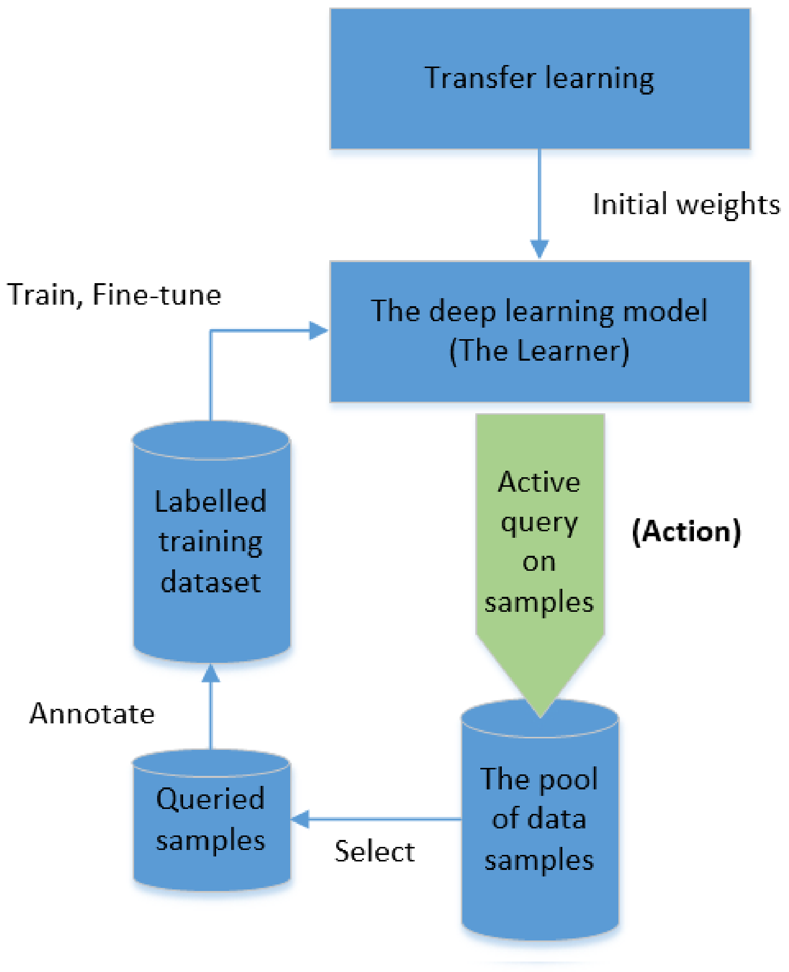

Figure 4.2: Data flow in the concept of the active learning process.

Chapter 6.

4.3 Auto-labeling

Auto-labeling is the final step in the design of the framework's architecture and is concerned with the auto-labeling capability. The initial classifier is based on the model outcome from the previous two stages. At this stage, we aim to evaluate the performance of the architecture using relevant metrics and test the capability of this framework using both the emission images and the DAGM dataset [24] for different levels of complexities in the image patterns. The DAGM dataset was mentioned earlier in Section 2.1.3 as being an an industrial optical inspection dataset provided by the German Association for Pattern Recognition. We will also discuss some considerations around the stopping criteria for the active learning stage where the auto-labeling begins.

4.4 Overview and Conclusions

In this chapter we have presented the key steps in the conceptual design of our framework including the way the model can be trained with a small amount of annotated data, the way active learning is used to determine which data samples need to be labelled to improve the model's training, and the value of automatic labelling to improve the size and quality of the annotated data.

To give an overview of the overall architecture for the proposed framework, a flow chart has been created as shown in Figure 4.3. This illustrates the flow of data (blue and purple arrows) and processing (black arrows); the corresponding numbered steps are listed and further explained below.

1. The total samples in the original dataset are separated into 2 groups namely labelled (the left route) and unlabelled (right route). Data flows along the blue arrows; The original dataset mentioned here is a general concept for all the image samples on which the method can be directly applied. They are already in the format of images that can be assigned labels. This is different from the raw data described earlier in Section 4.1 where the raw data were as 3D point clouds, a format that needs to be further processed into individual images and grouped into datasets. After this step, the dataset can be treated as the original dataset for labeling.
2. The initial training of the classifier uses the labelled data that has not yet been enlarged at this stage; data flows along the the purple arrow to the classifier in this step;
3. A group of data samples are selected out of the pool of unlabelled data according to the results of the active learning algorithms in this step;
4. The selected samples from the last step are passed to the (human) annotator to identify their proper label(s);

5. Samples labelled by the annotator flow to the collection of labelled dataset (purple box). This step enlarges the size of the labelled dataset to include manually annotated ones;
6. With the enlarged labeled dataset, fine-tuning is applied to the classifier to improve the deep representation and therefore the classification performance. This is also a step where the data flows from the labelled dataset to the classifier, following the same purple arrow as in step 2;
7. As a result of further fine-tuning in the last step, the performance of the updated classifier is examined. If the performance does not meet the stopping criteria of the active learning process (this could be a certain performance threshold or a certain number of labelled samples), the iteration will carry on from step 3. In this step there is no data flow, therefore the connection is indicated by black arrows;
8. If the performance is good enough, the resulting classifier will be used for automatically labelling (also referred to as pseudo-labeling) and the labelled data will further increase the size of the labelled training data.

We have now illustrated the design concepts around the overall architecture framework. Based on the content of this chapter, development and research work has been carried out on the relevant components as detailed in the next three chapters.

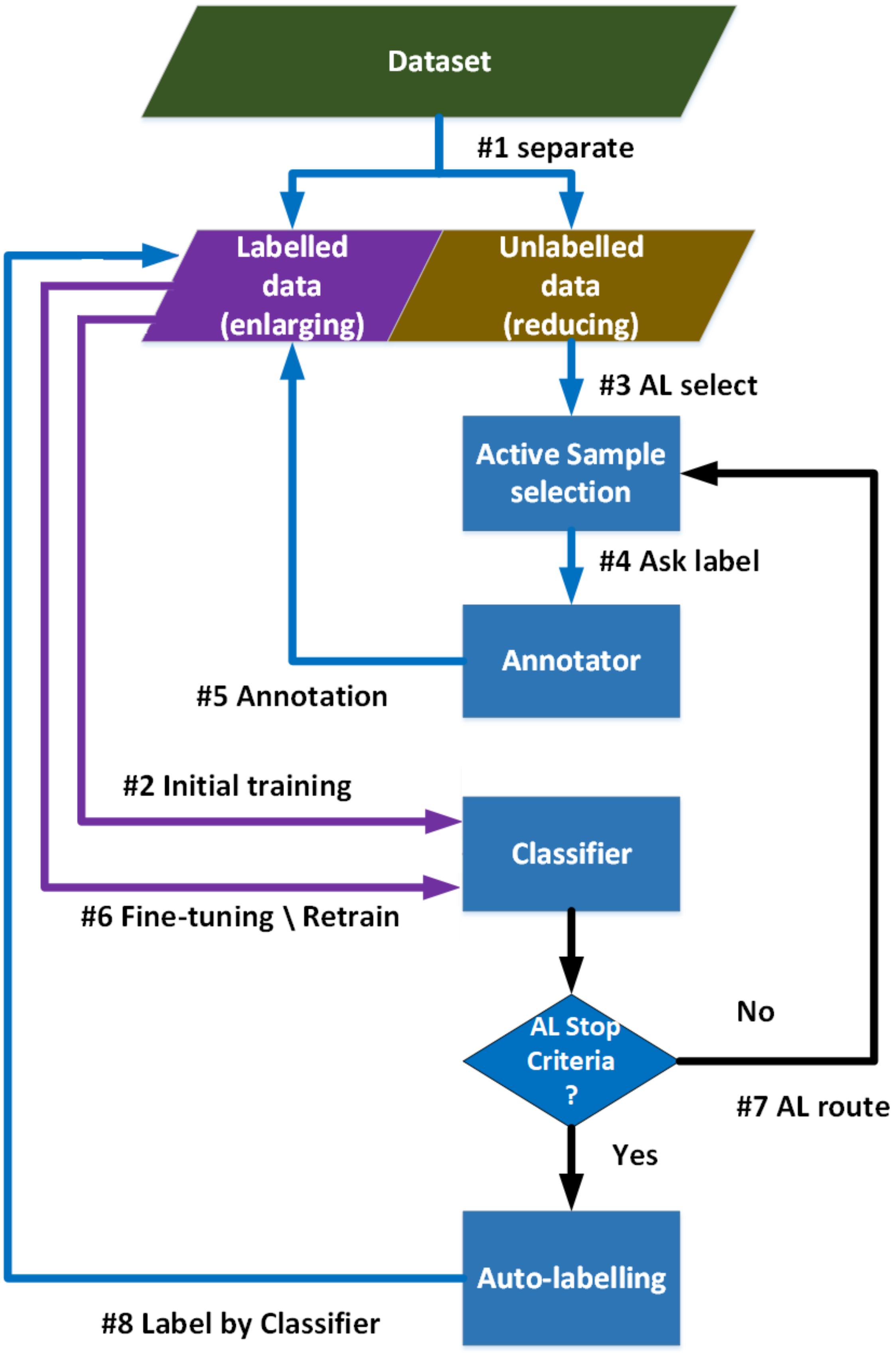

Figure 4.3: Overview of the architecture with data flow and processing steps.

Chapter 5

Defect Detection from Additive Manufacturing Emission Images

This chapter will illustrate the proposed classification method where we apply transfer learning with fine-tuning techniques to a deep learning model which is based on convolutional neural networks (CNNs). The classification method is tested using a combination of (a) an emission dataset in image format which was generated based on a 2D representation exported from the in-situ monitoring system as mentioned in the earlier chapters along with (b) the DAGM [24] dataset of images mentioned earlier in Section 2.1.3, for additional data support.

In this chapter, firstly, descriptions of the initial dataset and the experimental setup are provided in order to clarify the formation and arrangement of all the datasets used in the experiments. Secondly, the performance of two different types of classifiers is investigated: a Support Vector Machine (SVM) based classifier and a CNNs based deep learning model classifier, each used for detecting whether the input image is normal or has a defect in it. Thirdly, the chapter includes a section that illustrates further modifications to the architecture and layers of the CNN based deep learning model to improve classification accuracy, where all the datasets are processed by the improved model with a comparison of performance. Finally, the chapter also provides a list of specific parameters and settings used in the overall deep learning model and in the training process in order to support the reproducibility of the classification work.

In addition, this chapter further illustrates the approach used for the initial training and fine-tuning of a deep learning model, specifically focusing on the Classifier block shown earlier in Figure 4.3 in Chapter 4.

5.1 Creation of Initial Datasets

The first set of data to be used in our experiments was collected from the 2D representations of the emission images generated by the in-situ monitoring suite. The images represent the melt-pool conditions of the printed layers when printing a group of dog-bone shaped testing parts of Titanium alloy (Ti6Al4V) during the AM process. A total of 11,000 images were manually exported from the InfiniAM monitoring software as described in Chapter 4 of the thesis. Initially, all the raw image data are unlabelled and thus these raw data are not suitable to be used directly for training. The 11,000 sample images were manually inspected in order to select 150 images as defected samples and another 150 images as normal samples. These were labeled and used to create a dataset considered as the ground truth in the tests. The size of this labelled dataset is relatively small for the training of the machine learning models. We acknowledge that the manual selection of the samples could be better analysed and automatic methods could be used to select the most representative samples or investigate the impact of sample selection in the training process. However, in this investigation, we aim at demonstrating the feasibility of the approach with limited training data and we postpone the analysis of sample selection to future work.

As the currently available labelled data from the emission dataset are limited, we also used a well-known industrial optical inspection dataset provided by DAGM for testing purposes. The DAGM dataset contains 6 patterns of texture with each texture pattern containing 1,000 non-defective and 150 defective images, resulting in 6,900 images in 12 classes. Based on the original DAGM dataset, from each pattern, 150 non-defective sample images were selected from the original 1,000 non-defective samples to create new groups of testing datasets that involve 6 patterns and each pattern includes 150 defective and 150 non-defective samples. This dataset was used in conjunction with the manually labeled emission image dataset generated from the InfiniAM 3D model. In total, there are 7 patterns used in the experiments along with the 6 patterns from the DAGM dataset. The emission dataset is used as the 7th pattern and it contains samples of both defective and non-defective classes. Figure 5.1 shows a set of examples for normal and defective AM processing from each pattern. All the tests in the following sections are based on these datasets.

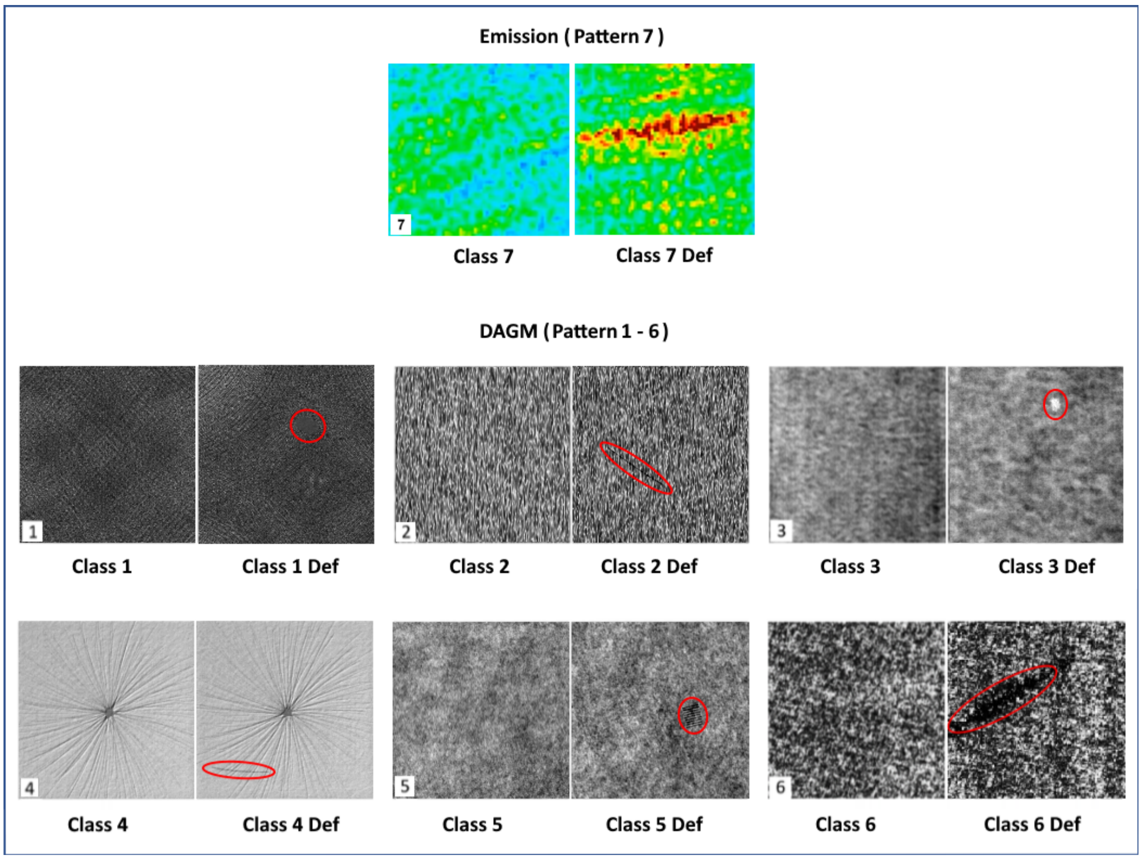

Figure 5.1: Sample images marked as Class 1-6 from the DAGM industrial optical inspection dataset [24]. The 6 texture patterns contain normal and defective in a particular location, marked with red ellipsoids. Images marked as pattern 7 are those we generated from the InfiniAM emission dataset.

5.2 Combining VGG-16 and SVM Classifiers

In this test, the image datasets are used as input for feature extraction. The features are extracted by the CNN architecture from a VGG-16 model using transferred weights trained using ImageNet data [6] without fine-tuning. Features are then passed to a Support Vector Machine (SVM) classifier to detect whether the input image is normal or has a defect. The architecture of this model is shown in Figure 5.2.

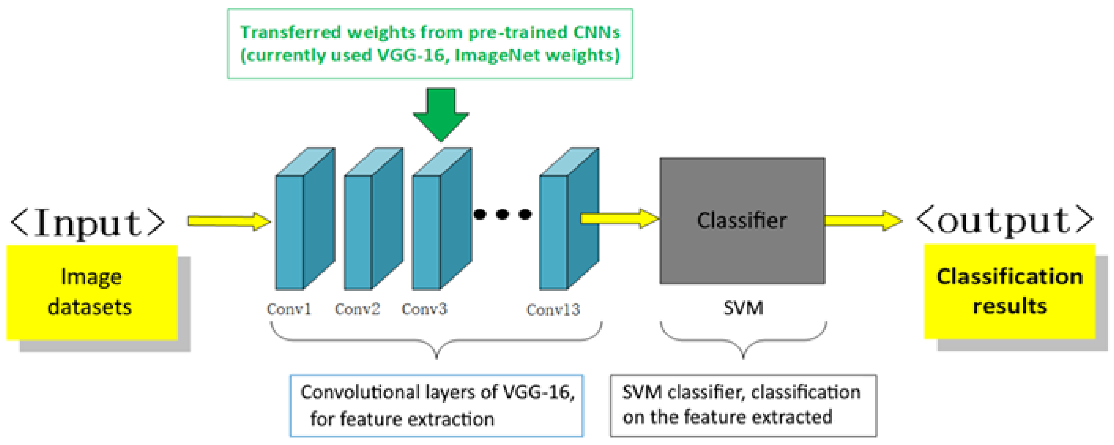
Transferred weights from pre-trained CNNs (currently used VGG-16, ImageNet weights)
<Input>
Image datasets
Conv1
Conv2
Conv3
Conv13
Classifier
SVM
<output>
Classification results
Convolutional layers of VGG-16, for feature extraction
SVM classifier, classification on the feature extracted

Figure 5.2: Architecture of the model using a combination of VGG-16 and SVM.

5.2.1 Hyperparameter Tuning for SVM Models

The SVM classifier divides the dataset into classes by creating virtual hyperplanes in a multi-dimensional space. The SVM classifier in this paper is implemented using the C-Support Vector Classification functions from the Python ML package Scikit-learn [133]. The hyperparameters used for the SVM classifier are the type of kernel, the regularisation parameter C and the kernel coefficient Gamma. Table 5.1 lists the hyperparameters used in our experiments, and a brief description of what these parameters are. In addition, some of the commonly used kernels are summarised below:

1. Linear Kernel (linear): This kernel is used for linear separation of data.

2. Radial Basis Function Kernel (rbf): Also known as the Gaussian kernel, it is used for non-linear separation.

3. Polynomial Kernel (poly): This kernel is used when the data is not linearly separable and transforms the data into higher-dimensional space using polynomial functions.

4. Sigmoid Kernel (sigmoid): This kernel is often used for neural networks and is similar to the sigmoid activation function.

Table 5.1: Parameters for the SVM classifier.

Name of Hyperparameter	Type or Value	Description
Kernel	rbf, linear, poly, sigmoid	The main function of the kernel to transform the given input data into the required form.
Regularisation parameter C	0.1, 1, 10, 100, 1000	C is the penalty parameter, which is used to maintain regularisation. It represents misclassification or error term of the SVM classifier.
Gamma	1, 0.1, 0.01, 0.001, 0.0001	The kernel coefficient for the RBF, the value of Gamma depends on the number of features from the data

The grid search function from Scikit-learn is used to conduct the tuning of the hyperparameters for the SVM model. For a further illustration, this is a method provided by scikit-learn for hyperparameter tuning in machine learning models. It performs an exhaustive search over a specified parameter grid to determine the best combination of hyperparameters for a given model. This technique is useful for optimising model performance by selecting the hyperparameters that result in the highest cross-validated score. The following steps illustrious the process of tuning:

1. Define a grid of hyperparameters: The grid is defined as Kernel:[rbf, linear, poly, sigmoid]; C: [0.1, 1, 10, 100, 1000]; gamma: [1, 0.1, 0.01, 0.001, 0.0001].

2. Then the grid search function trains and evaluates the model using each combination of hyperparameters in the grid. The performance of each model configuration is evaluated using a 5-fold cross-validation.

3. After evaluating all combinations of hyperparameters, the best combination is selected according to the best cross-validated score. this is automatically done by the grid search function from Scikit-learn. In this way, the model with the best combination of hyperparameters is obtained.

5.2.2 Classification Performance of the SVM Model

During the training and validation process, the 7 patterns were individually passed through the model. As illustrated previously, the dataset includes 300 samples for each pattern, which are further divided into 2 classes: 150 positive samples and 150 negative samples. 70% of the samples were used for training and the remaining 30% were used for validation. After the grid search and training, the classification results from the best model for each pattern and the best combination of hyperparameters are shown in Table 5.2

Table 5.2: Classification results from an SVM classifier with hyperparameters

Pattern	Class	Precision	Recall	F1-score	Accuracy	Kernel	C	Gamma
Pattern 1	Defect	0.60	0.71	0.65	0.64	rbf	1000	0.1
	Normal	0.69	0.58	0.63				
Pattern 2	Defect	0.78	1.00	0.88	0.88	poly	0.1	1
	Normal	1.00	0.78	0.87				
Pattern 3	Defect	0.72	0.74	0.73	0.74	linear	100	1
	Normal	0.75	0.73	0.74				
Pattern 4	Defect	0.83	0.76	0.79	0.82	poly	1	0.1
	Normal	0.81	0.87	0.84				
Pattern 5	Defect	0.86	0.92	0.98	0.88	rbf	1000	0.1
	Normal	0.90	0.82	0.86				
Pattern 6	Defect	0.88	0.79	0.83	0.83	rbf	100	0.1
	Normal	0.79	0.88	0.83				
Pattern 7	Defect	0.97	0.95	0.96	0.96	poly	0.1	1
	Normal	0.94	0.97	0.96				

As the size of each labeled dataset is relatively small, it is remarkable to observe how, even with limited training data, the classification results for certain patterns, such as pattern 7, exhibit relatively high accuracy rates, achieving an overall classification accuracy of about 96%. This is followed by pattern 2 (88%), pattern 5 (88%), pattern 6 (83%), and pattern 4 (82%). Despite these positive outcomes, the approach still demonstrates relatively lower performance on pattern 1 (64%) and pattern 3 (74%).

There are three possible reasons to explain such low performance. First, the number of samples for training is not sufficient to train the SVM classifier; second, some of the features extracted from the datasets may not be representative enough and may contain too much unnecessary information; third, SVM may not be a suitable classifier for certain patterns in the test data. To address these issues and try to improve results, we propose and test a different approach based solely on CNNs, which relies on the VGG-16 architecture extended and modified for our specific use case.

5.3 Extending VGG-16 for Classification Using Dense Layers

This approach to classification relies on a transfer learning method in which the 13 convolutional layers from the pre-trained convolutional layers of VGG-16 from the previous model are still used for feature extraction and the weights in these layers are unchanged. To modify the classifier, the SVM classifier is replaced by fully connected layers. As illustrated in Figure 5.3, after the convolutional layers, 2 dense layers with ReLU activation function are added and followed by 1 dense layer as the output layer using Sigmoid as the activation function, since detecting normal or defect individually for each pattern is a binary classification task.

In this experiment, rather than directly testing on all the image groups, three patterns which are considered to be the most challenging, the average and the easiest according to the earlier results, are selected for further exploration. As illustrated in Table 5.2, the selected patterns are: pattern 6, with average performance in the previous test; pattern 4, with one of the worst performances in the previous test;

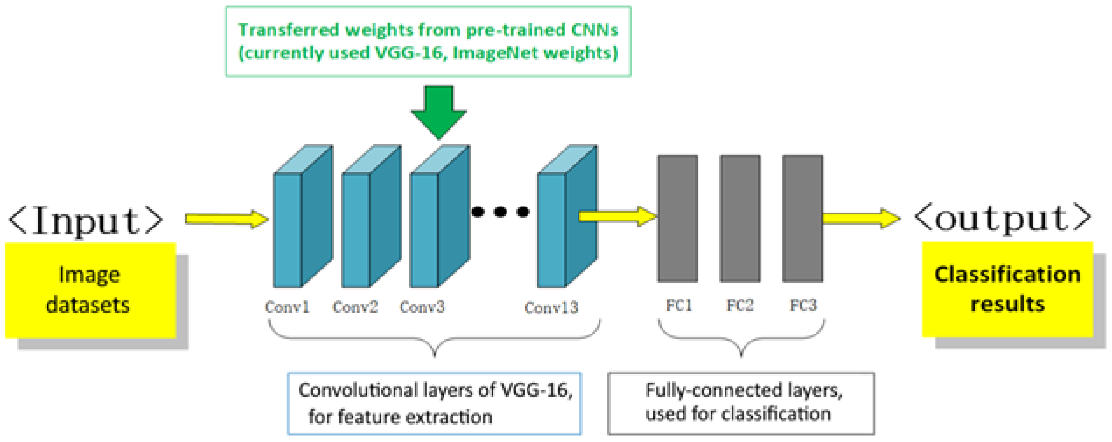
Transferred weights from pre-trained CNNs (currently used VGG-16, ImageNet weights)
<Input>
Image datasets
Conv1
Conv2
Conv3
Conv13
FC1
FC2
FC3
<output>
Classification results
Convolutional layers of VGG-16, for feature extraction
Fully-connected layers, used for classification

Figure 5.3: Architecture of the model based on VGG-16 and fully connected layers.

pattern 7, with the best performance in the previous test.

5.3.1 Testing Pattern 6 with Average Performance

The initial investigation of the performance of this new model began with the dataset from pattern 6 (refer to Figure 5.1) for binary classification and the length of training in this test was set to 200 epochs. Accuracy and loss are the metrics used to trace and evaluate the training and validation process. Figure 5.4 shows the curves for accuracy and loss for both training and validation for pattern 6.

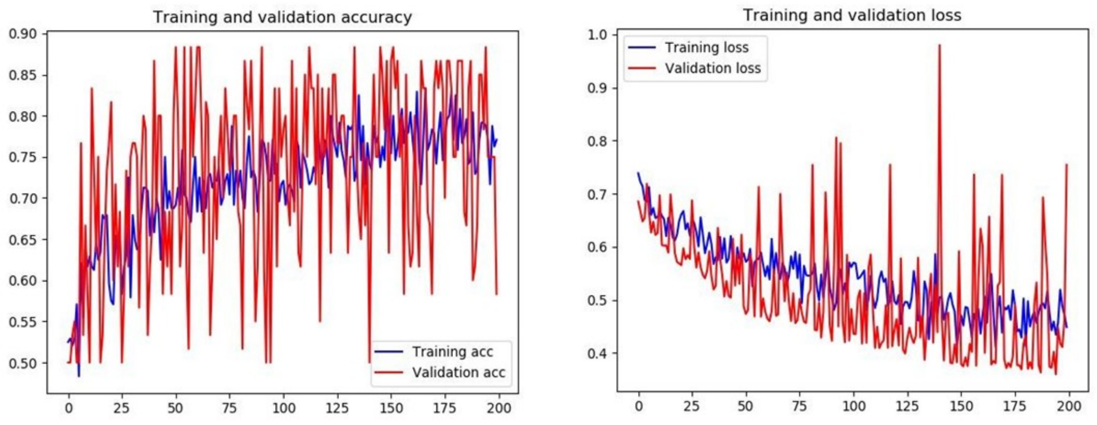

Figure 5.4: Accuracy and loss over 200 epochs (the horizontal axis shows the number of epochs) with transfer-learning and without fine-tuning. The vertical axis in the left chart represents accuracy while the vertical axis in the right chart represents loss.

In Figure 5.4, the lines show an overall increase in accuracy (the left image) and a decrease in loss (the right image) both for training and validation but with significant fluctuation. The results do not show overfitting as the validation loss is not significantly larger than the training loss. Regarding the problem of extremely high fluctuation in the validation, the following points are relevant:

1. The number of samples for the test is relatively small and as the loss is still relatively high and unstable, even a small number of classification results will cause major changes in the overall accuracy;

2. There may be too much noise considered as features. To address this problem, adding a pooling layer between the convolutional and the dense layers could be a solution;

3. According to Kim *et al.* (2017) [22] with a frozen network for the convolutional layers, the results should have a level of accuracy in the range of 78%-85%. This means that using the setup without fine-tuning, the fluctuation should be around 80%. Fine-tuning has therefore been applied as discussed in the next sub-section to reduce the fluctuation and improve the overall classification accuracy.

To further improve the performance of the model, an average pooling layer was added between the convolutional layers and the dense layers to reduce the spatial dimension of the output from the convolutional layers. We performed a test by training a new model for only 50 epochs to show that the modified architecture can effectively and quickly reduce the fluctuation for accuracy and loss in both training and validation, as shown in Figure 5.5.

After improving the model with the addition of a pooling layer, fine-tuning was then applied in order to increase classification accuracy and reduce the loss. Based on this new configuration, the last 3 convolutional layers from the VGG-16 architecture are unlocked for fine-tuning. The weights in the unlocked layers are modified by the training replications using the image datasets. Using the updated model and pattern 6, the training and validation are re-executed for 200 epochs to ensure the process reaches convergence. The curves for training and validation are shown in Figure 5.6.

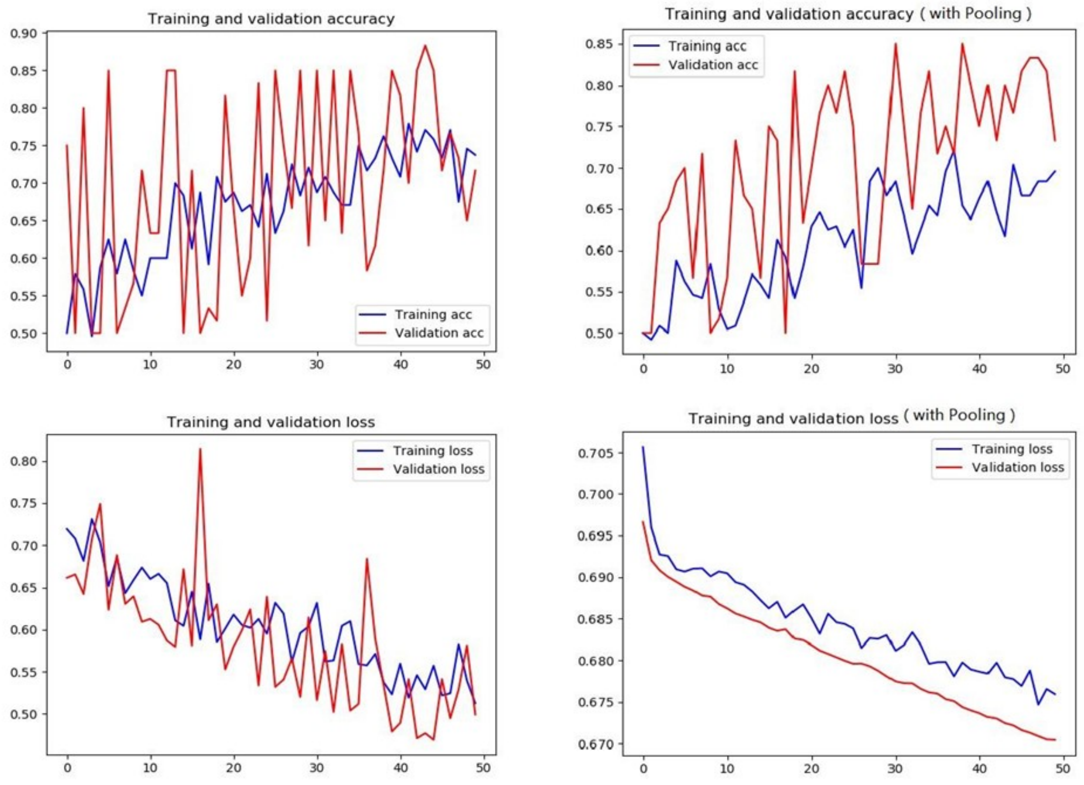

Figure 5.5: Reduction of fluctuation in the curves of overall accuracy and loss after adding the pooling layers (top-right and bottom-right) compared to results without pooling a layer (top-left and bottom-left).

Around epoch 200, the training accuracy is over 97% while the validation accuracy is around 95%. The training loss starts to converge at about the 100th epoch with a value around 0.1 while the training loss continues to drop slightly.

5.3.2 Testing Pattern 4 with Worst Performance

Similar tests were also applied on pattern 4 and with the updated setup the validation accuracy at about the 200th epoch is also over 96% with a relatively stable value of the validation loss of 0.1 (refer to Figure 5.7). This result is significantly more accurate than the one obtained from the SVM classifier in the previous experiment, where the model was not able to identify the classes for pattern 4.

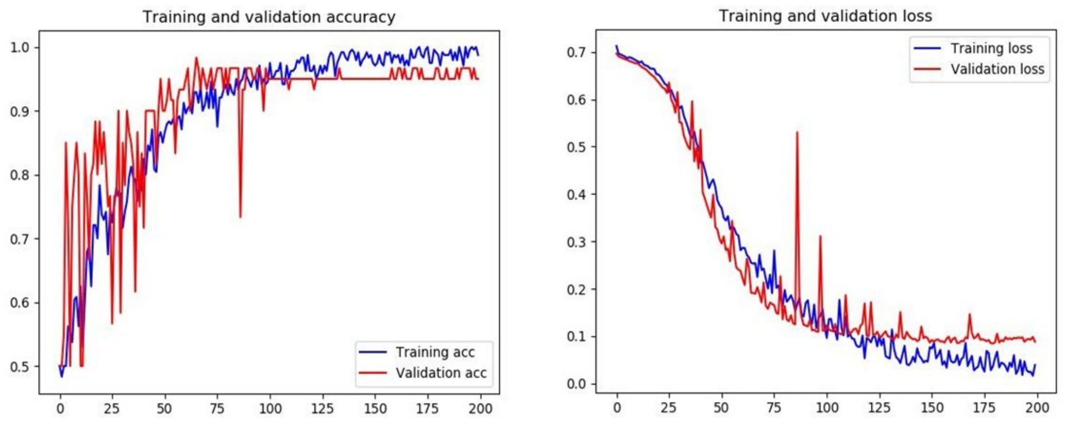

Figure 5.6: Accuracy and loss for pattern 6, using a model with pooling and fine tuning on the last 3 convolutional layers.

5.3.3 Testing Pattern 7 with Best Performance

Pattern 7 is the emission dataset of images generated from the 2D presentation collected and converted from the InfiniAM in-situ monitoring system. This dataset is the one we specifically collected for this research. The model is trained using the dataset for pattern 7 for 200 epochs. The output of the classification shows considerably high values in accuracy for both training and validation. As shown in Figure 5.8, the training process reaches convergence after around the 130th epoch with a training accuracy around 99% and a validation accuracy over 98%.

5.3.4 Testing On All Patterns

The results of the classification by the model we introduced with fine-tuning has shown significant improvement in overall accuracy across different patterns. To complete this investigation, the remaining patterns 1, 2, 3 and 5 were also tested individually using our modified VGG-16 architecture with transfer learning and fine-tuning. Results for all 7 patterns are shown in Table 5.3, including the average values (after convergence) of training accuracy, training loss, validation accuracy, validation loss and the training duration (in epochs) for each corresponding pattern.

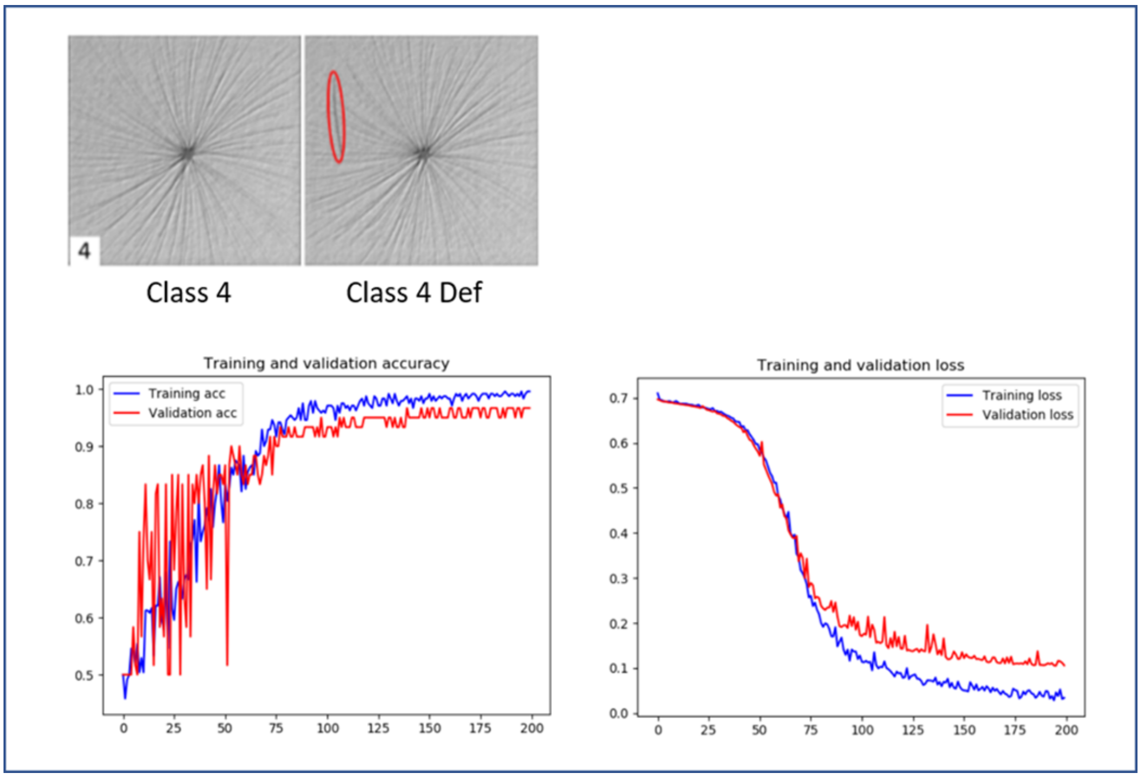

Figure 5.7: Accuracy and loss for pattern 4 using a model with pooling and finetuning on the last 3 convolutional layers.

5.4 Parameters and Settings

We now present the values of the parameters used in the overall model for defect classification, as well as settings for the training process. The deep learning model illustrated in this chapter is implemented using the Python deep learning package Keras. Figure 5.9 describes the architecture of the model layer by layer, and the shape of input/output tensors in each layer. Table 5.4 contains a listing of the hyperparameters used for the training of the model.

Stopping criterion: In the tests, the numbers of epochs set for the stopping criterion were simply set to be large enough for the curves of validation accuracy and loss to reach convergence and where the curves would not have any significant change with a longer duration of training. Because the input dataset is different in each test, the number of epochs required will also be different. For example, to reach convergence in the training process the tests for dataset 1 requires more epochs than that required for the test of dataset 7.

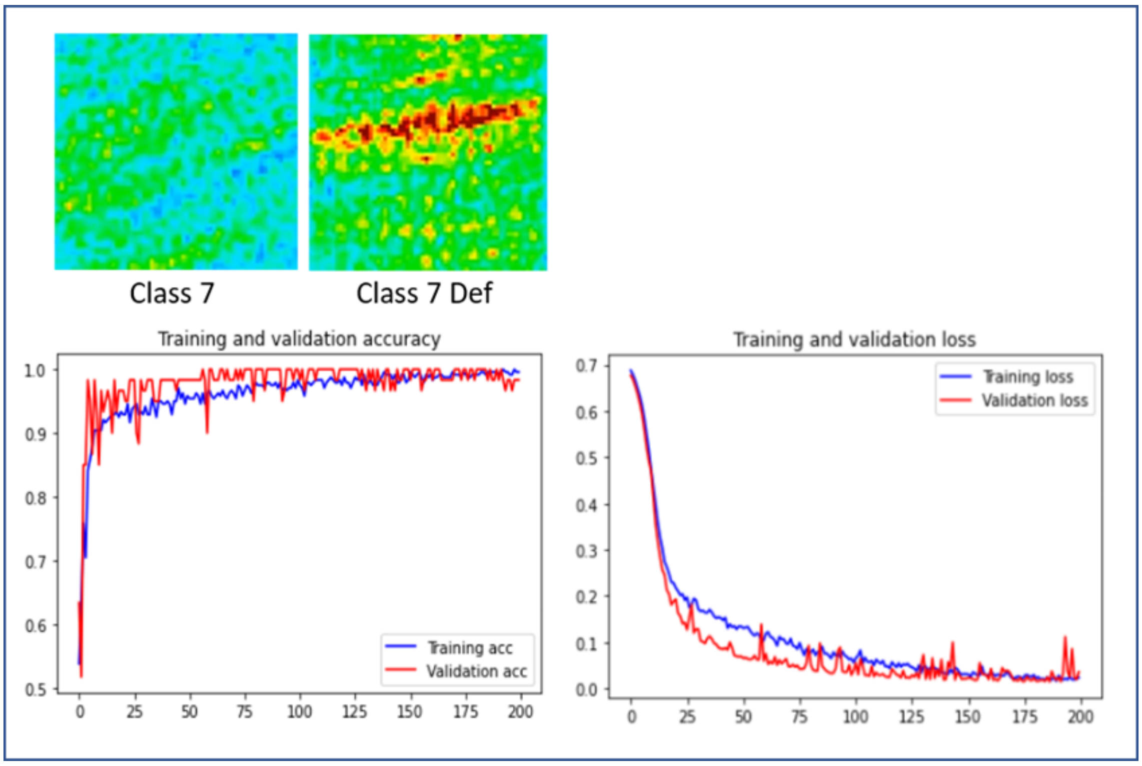

Figure 5.8: Accuracy and loss for pattern 7 using the model with pooling and fine tuning on the last 3 convolutional layers.

Pooling layer: To reduce the spatial dimension of the output of the feature extraction model based on VGG-16, a Global Average Pooling 2D (GAP2D) layer was added between the outcome of the VGG-16 model used for feature extraction and the dense layers of the classifier. The GAP2D layer receives the output tenser from the VGG-16 model and applies a global average pooling operation for spatial data. By doing this the number of total channels in the output of the GAP2D layer is reduced to 512 and makes it ready to be processed in the dense layers. This is shown in Figure 5.9.

Dense layers: We added three dense layers (also known as fully connected layers) to our model. The activation function used in the first two layers is the ReLU activation function, while a Sigmoid activation function is used in the third dense layer for binary classification.

Table 5.3: classification results from the modified VGG-16 architecture with transfer learning and fine-tuning

Patterns	Avg. Training Accuracy	Avg. Training Loss	Avg. Validation Accuracy	Avg. Validation Loss	Training Duration (epochs)
Pattern 1	0.98	0.03	0.96	0.09	800
Pattern 2	0.99	0.02	0.98	0.03	400
Pattern 3	0.98	0.04	0.96	0.09	600
Pattern 4	0.97	0.05	0.96	0.10	200
Pattern 5	0.99	0.02	0.98	0.04	400
Pattern 6	0.97	0.05	0.95	0.09	200
Pattern 7	0.99	0.03	0.98	0.03	200

Table 5.4: Hyperparameters used for training the deep learning model.

Name	Type / Value	Description
Optimiser	Stochastic Gradient Descent (SGD)	Optimisers are used to change the attributes of the neural network to reduce the losses
Loss function	Binary cross entropy	The loss function computes the quantity that a model should seek to minimise during training
Learning rate	0.001	The step size at each iteration while moving toward a minimum of a loss function during the training
Evaluation metric	Accuracy, Loss	The metric is a function to judge the performance of the model

5.5 Discussion

From the results obtained it is clear that the deep learning model introduced in our approach which leverages transfer learning and fine-tuning has relatively good classification performance on all 7 patterns in the initial dataset. For certain patterns, such as pattern 5 and pattern 7 the results show higher values in accuracy even before any optimisation. More importantly, by replacing the SVM with fully connected layers as a classifier, the approach allows the overall model to be fine-tuned using a relatively small amount of data to further adjust the weights in both the convolutional and fully connected layers for low performance patterns. The application of fine-tuning greatly improved the accuracy in the classifications for those

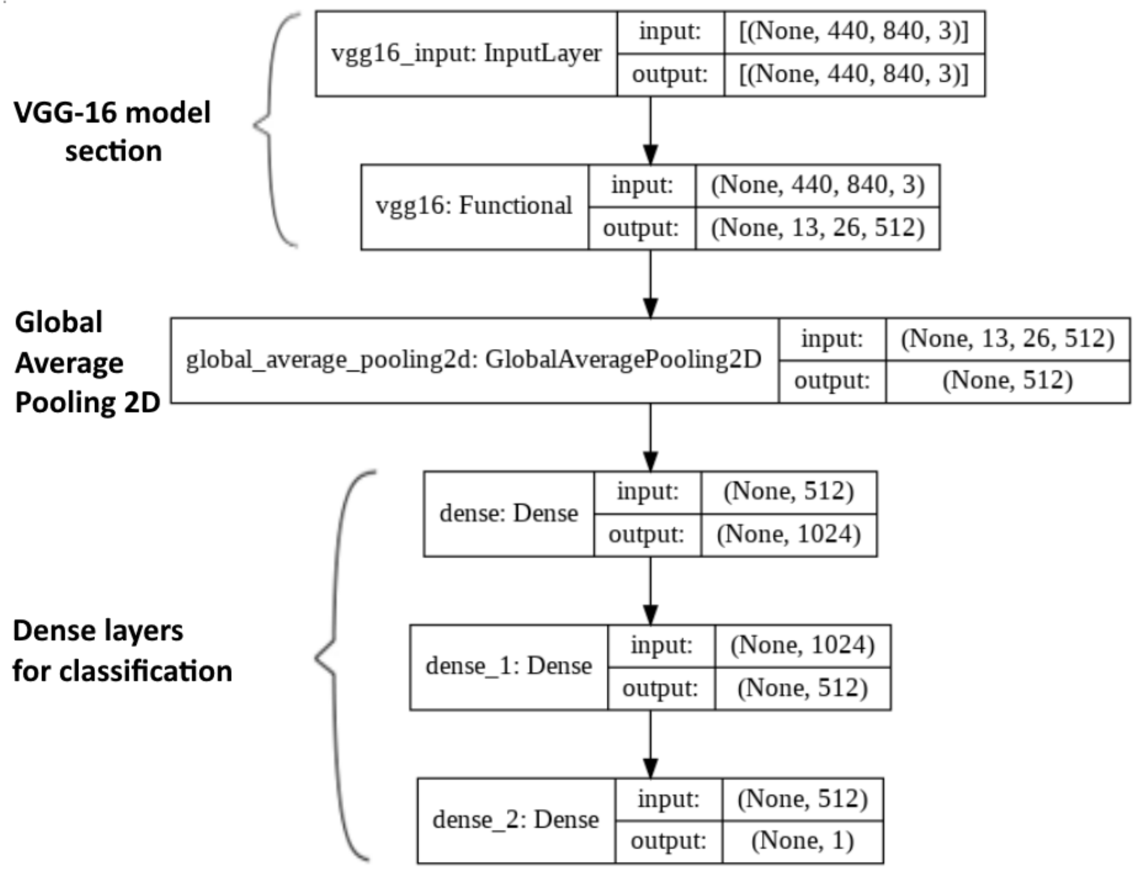
VGG-16 model section
vgg16_input: InputLayer
input: [(None, 440, 840, 3)]
output: [(None, 440, 840, 3)]
vgg16: Functional
input: (None, 440, 840, 3)
output: (None, 13, 26, 512)
Global Average Pooling 2D
global_average_pooling2d: GlobalAveragePooling2D
input: (None, 13, 26, 512)
output: (None, 512)
Dense layers for classification
dense: Dense
input: (None, 512)
output: (None, 1024)
dense_1: Dense
input: (None, 1024)
output: (None, 512)
dense_2: Dense
input: (None, 512)
output: (None, 1)

Figure 5.9: Overall model including the VGG-16 architecture layers, the GAP2D layer and the dense layers for classification. The value "None" in the shape of tensors indicates the dimension is a variable.

patterns and it is also expected to allow the model to be continuously improved as the available datasets grow in size.

With the aid of transfer learning, this approach greatly reduces the required amount of labelled data and computational power needed for training. In fact, manually creating large training databases is time consuming, expensive, and often infeasible in industrial production settings. The lack of properly labelled data is a common issue for the application of ML, especially in AM processes.

5.6 Experiments on Images from Additive Manufacturing

In the previous sections we illustrated the feasibility of transfer learning and fine-tuning by applying relevant techniques to a convolutional neural network based deep learning model. Following that we performed classification related tasks on an industrial dataset, the DAGM dataset and on our own emission images dataset. So far, all the work that has been done has been focused on the study, investigation and experiments on the task of classification.

Although classification methodologies can be developed into effective approaches for various applications in real industrial setups such as monitoring of the printing process of additive manufacturing, in certain circumstances the object of interest that is to be verified will require further detailed and higher level information which can be beyond the scope of classification techniques. This occurs when the techniques of object detection take over the work in order to provide desired properties and detailed information about objects detected in the emission images in order to support subsequent analysis, control and modification tasks. In this section, we test several approaches for object detection and segmentation using computer vision including the conventional approaches as well as some of the recent deep learning based approaches.

5.6.1 Object Detection Using Conventional Approaches

This section aims to investigate the performance of some of the conventional approaches to object or blob detection from images generated during the additive manufacturing process. In this we use LoG, Difference of Gaussian (DoG) and DoH techniques. For illustration of the different techniques we use a sample colour image which is an original image before it is then converted to grayscale. We show the colour version as the RGB image is much easier for manual observation of the stages of processing. The colour version of the image is shown in Figure 5.10.

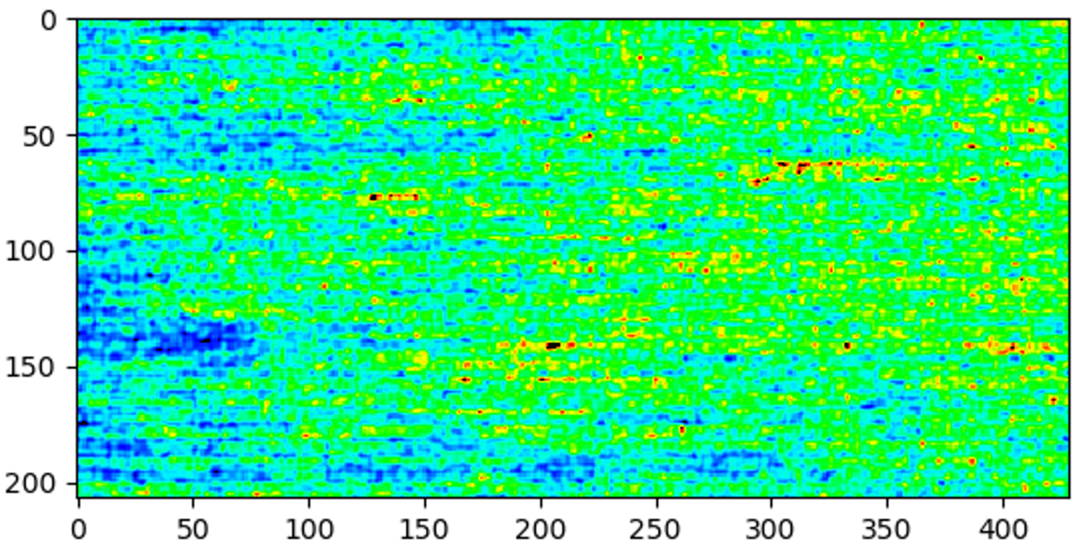

Figure 5.10: Sample image used for illustrating blob detection via conventional approaches.

To further process the image, it was converted to grayscale as shown in Figure 5.11 and then we applied a threshold of the mean intensity value of the total of all pixels and this is shown in Figure 5.12. This threshold may not be the best solution, but here it is just used for testing purposes for blob detection.

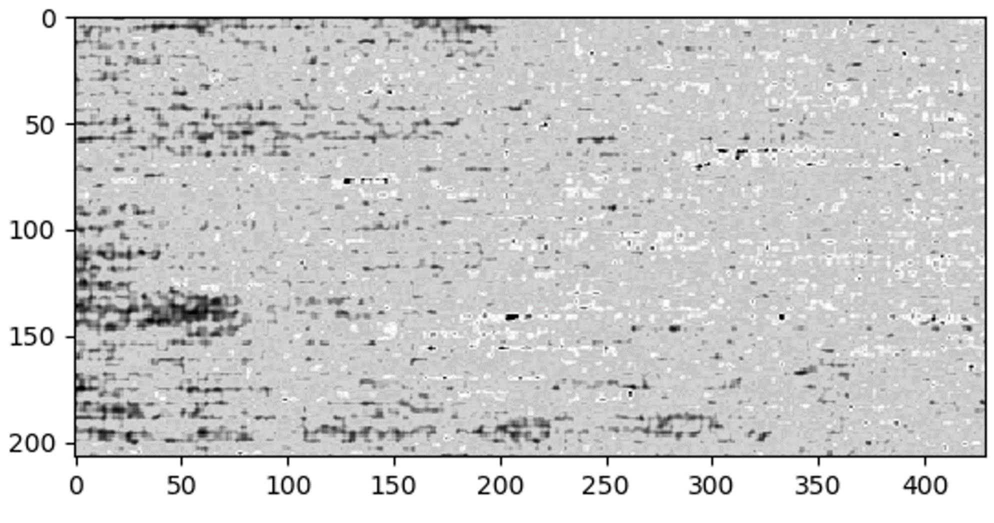

Figure 5.11: Greyscale version of image used to illustrate performance of algorithms.

For blob or object detection, the scikit-image Python package was used. To investigate the performance of different algorithms, blobs are detected using 3 algorithms and for illustration, all blobs are marked using red circles. This is represented as a triple (x, y, r) where x and y are the position of the centre of the detected blobs

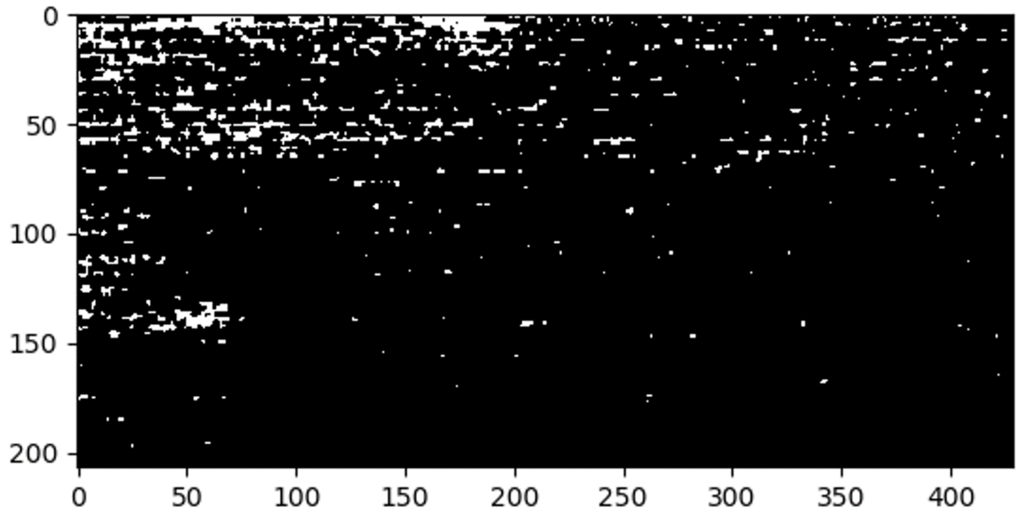

Figure 5.12: Result after applying thresholding on the greyscale version of image.

and r is the radius as illustrated in the following subsections.

Testing using Laplacian of Gaussian (LoG)

This is the most accurate and slowest approach. It computes the Laplacian of Gaussian images with successively increasing standard deviation and stacks them up in a cube. Blobs are the local maxima in this cube. Detecting larger blobs is especially slower because of larger kernel sizes during convolution. Only bright blobs on dark backgrounds are detected. The output is shown in Figure 5.13.

Testing using Difference of Gaussian (DoG)

This is a faster approximation compared to the LoG approach. In this case the image is blurred with increasing standard deviations and the difference between two successively blurred images are stacked up in a cube. This method suffers from the same disadvantage as the LoG approach for detecting larger blobs. Blobs are again assumed to be bright on dark and the output on the sample image is shown in Figure 5.13.

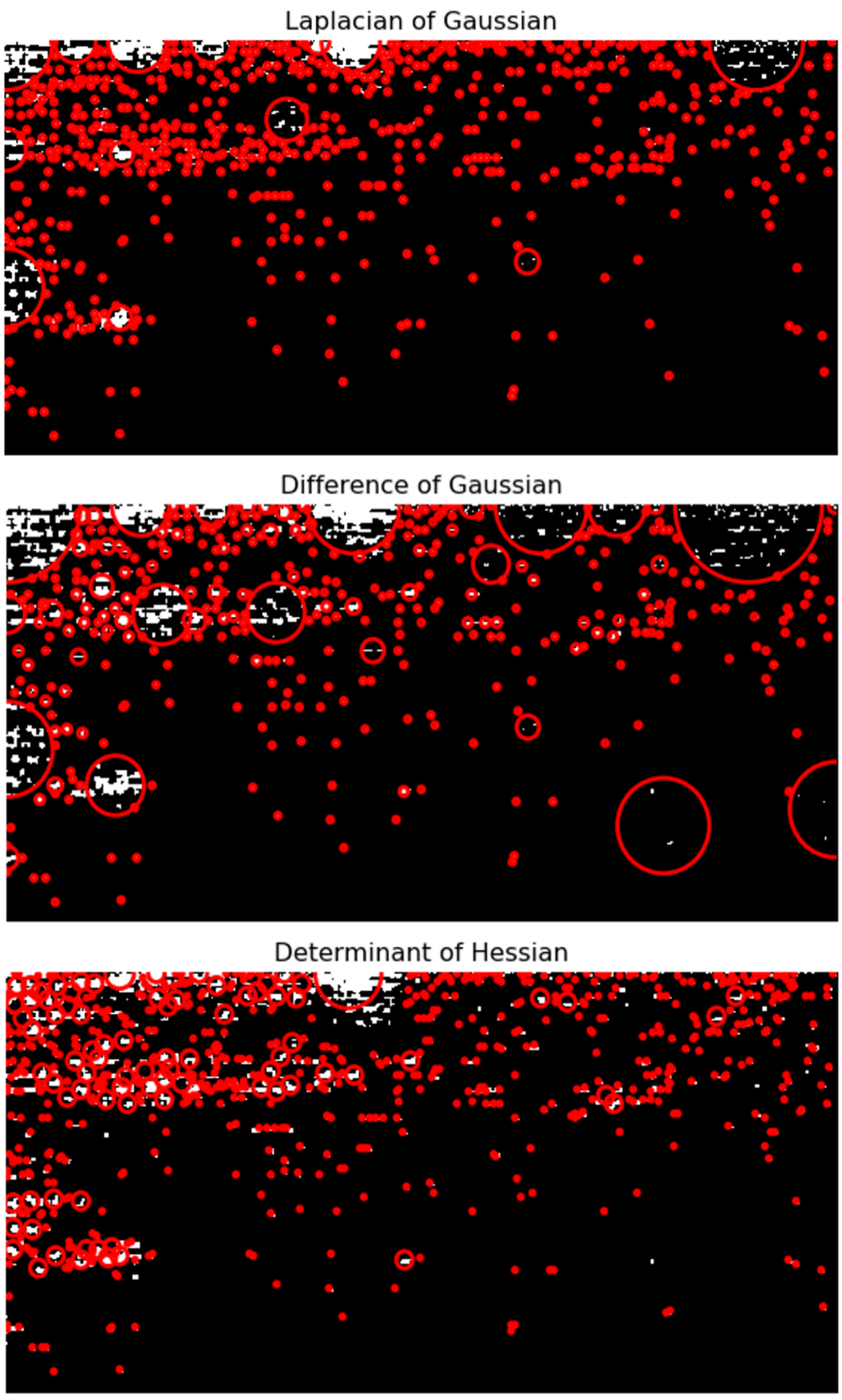

Figure 5.13: Comparison of the performance of different algorithms for blob detection

Tesing using Determinant of Hessian (DoH)

This is the fastest approach. It detects blobs by finding maxima in the matrix of the Determinant of Hessian of the image. The detection speed is independent of the size of blobs as internally the implementation uses box filters instead of convolutions.

Bright on dark as well as dark on bright blobs are detected. The downside of DoH is that small blobs (smaller than 3 pixels) are not detected accurately. The output on the sample image, along with the output from the other 2 approaches, is shown in Figure 5.13.

5.6.2 Discussion on Conventional Approaches to Blob/Object Detection

To further improve the detection performance using conventional approaches we can consider three points. First, we can apply filters to reduce noise in the image before applying the filtering threshold. Second, we can find a more appropriate threshold to better segment the grayscale image and thirdly we can select the most suitable algorithm for blob detection tasks. In practise all these improvements can be time-consuming to evaluate multiple detection algorithms and to compare their performances.

Furthermore, as shown in the illustrating images in Figure 5.13, there is a common issue across the three algorithms we used that the blob detection algorithms do not only consider some connected blobs as a single one, but also group unconnected close blobs into a single circle/blob which can be inaccurate. This motives us to proceed on the investigation of an approach to blob detection with higher capabilities, namely the Mask R-CNN which is one of the state-of-the-art deep learning techniques on object detection.

5.7 Testing Using Mask R-CNN

In this investigation, we will investigate the performance of Masked R-CNN on sample images that contain microstructure defects collected from additive manufacturing.

5.7.1 Source Images

For the initial study, a series of images that contain microstructure defects collected from a group of 3D printed parts manufactures using Al-7Si-0.3Mg alloy have been supplied to us. The situation in this study is similar to our previous research: the samples contained in the images are limited in number and the certain objects in the microstructures that we are searching for in each sample are also scarce. Considering this situation, we decided to leverage the advantage of Mask RCNN, transfer learning and fine-tuning rather than training from scratch in order to overcome the issue of potentially insufficient training data.

The original source of the data are 15 images captured by camera from 3D printed Al-7Si-0.3Mg samples. The shape of these samples, the objects being manufactured, are cubes with a dimension of 5mm $\times$ 5mm by 5mm. To investigate the microstructures in the samples, etching treatment has been applied to selected surfaces of the samples where the samples were etched with Keller's reagent (HF, HCl, HNO3 and water) by an immersion method and holding for 25-30 seconds. After etching the samples were cleaned with water and dried with a hot air hair dryer.

Figure 5.14 shows an example of the surfaces of the layers in sample 1 from the image set. Under 5$\times$ magnification, the left image shows a section of the surface of the layer before etching and the right image shows the appearances of the microstructures in a section area of the layer surface after etching. It should be noted that the sections shown in the left and right images in Figure 5.14 do not correspond to each other due to the limits of the original image set. However, the images do illustrate the effect of the etching treatment in order to significantly expose the microstructures that were previously covered under the surface, and the image on the left is used to illustrate this concept. The interesting objects for the purpose of our research are only present in the images of the surface layer after etching.

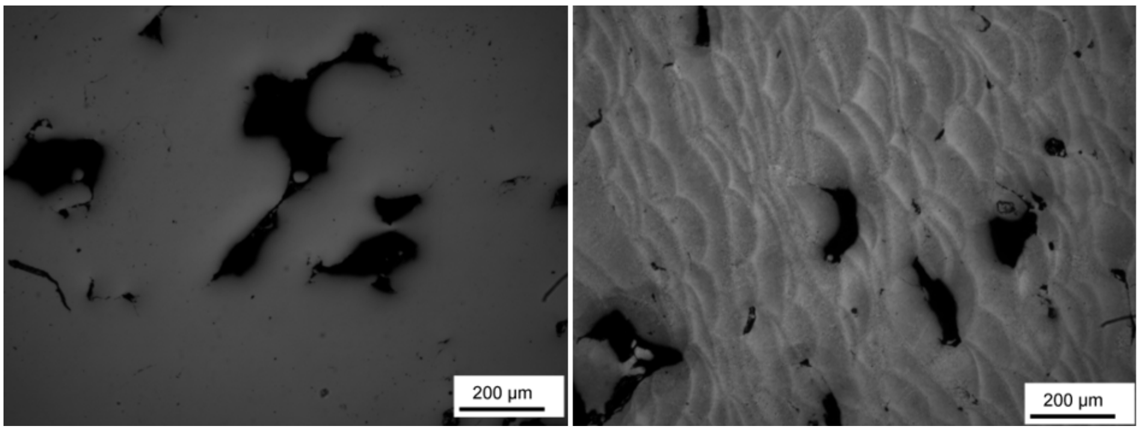

Figure 5.14: Appearances of the surface of various sections of the layer from sample 1, before etching (left) and after etching (right) under a 5× magnification.

5.7.2 The Microstructure Objects

In general, there are two types of microstructures contained in the images namely melt-pools and void spaces. Figure 5.15 shows an example of these relevant objects. In this research, the object that we focus on is the melt-pool as a target for applying the deep learning method for object detection. There are two reasons for this decision:

1. From the aspect of geometry, in this set of images the shapes of the melt-pools are in a general pattern that potentially can be learned by a deep learning model while the shape of the void spaces can be quite variable and may not follow general patterns.

2. When considering availability, the number of samples of the melt-pools that can be segmented to created training dataset for deep learning based object detection are far larger in number than the number from the shapes of the void spaces. Thus, in comparison to the void spaces, it is more feasible to chose melt-pools as the object to be detected in this work.

5.7.3 Creation of Training Data

To create a dataset for the training of a deep learning model, we divide a single image that contents the shapes of melt-pools into smaller sub-images such that each sub-

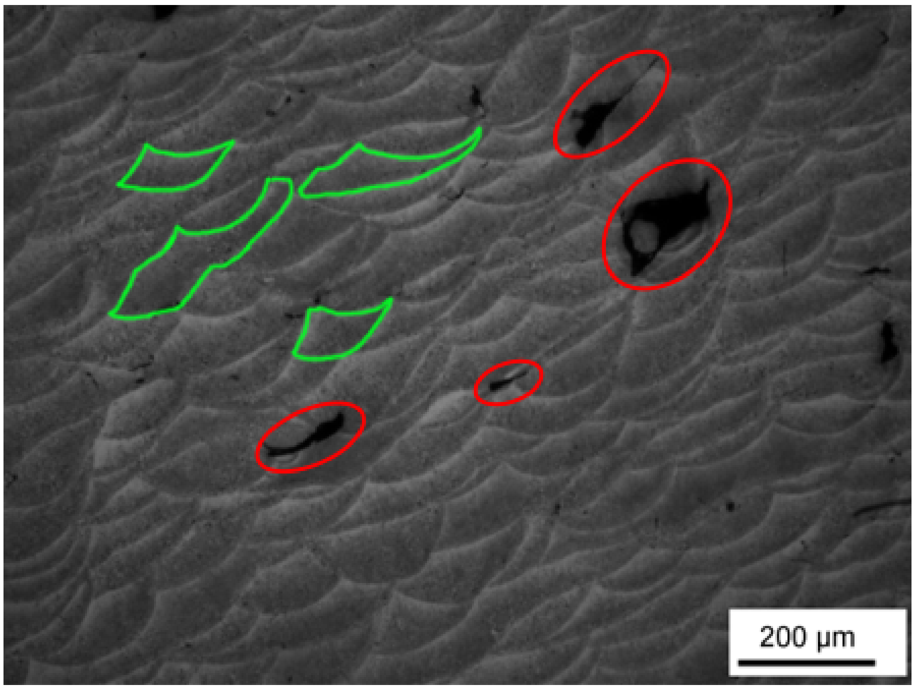

Figure 5.15: Examples of melt-pools (green marks) and void spaces (red marks).

image involves several samples of melt-pools for a total of around 40 sub-images. For the annotation work, an online tool called makesense.ai [134] for annotation has been used to annotate the melt-pool as objects in the sub-images using a polygon. An example of such an annotation is shown in Figure 5.16. The annotation information is saved in COCO JSon format to match the input requirement of the specific Mask R-CNN model that will be described in the next section.

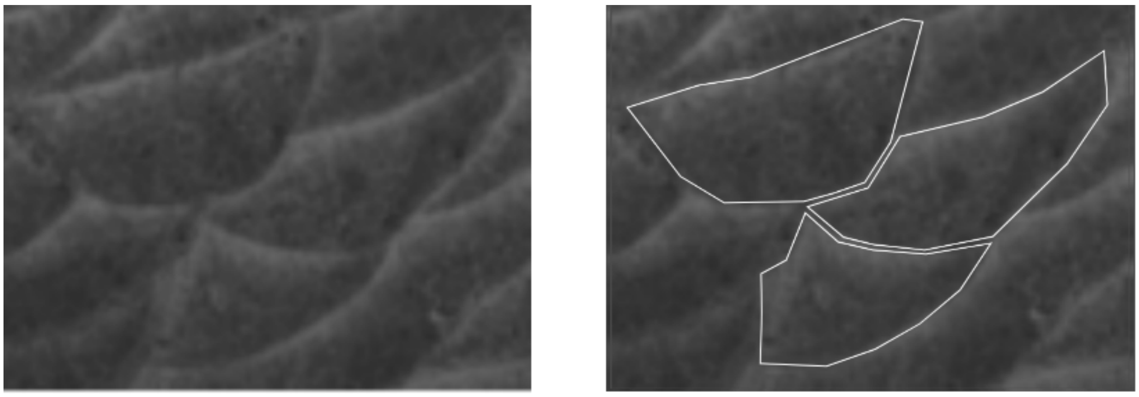

Figure 5.16: Examples of annotations on the melt-pools as objects to be detected.

5.7.4 Using the Mask R-CNN Based Model

The Mask R-CNN based model is mainly based on the Mask R-CNN for Object Detection and Segmentation that was developed by Matterport in [135]. The implementation of this Mask R-CNN on is in Python 3 as well as the Keras deep learning packages with TensorFlow used as the back-end. This model can detect instances of interesting objects and generate bounding boxes and segmentation masks for each instance in the input image. It is based on the Feature Pyramid Network (FPN) [136] and uses a ResNet101 [137] backbone. As further explanation, FPN combines low-resolution, semantically strong features with high-resolution, semantically weak features via a top-down pathway and lateral connections. This feature pyramid has rich semantics at all levels and is built quickly from a single input image at scale and without sacrificing representational power, speed, or memory. The model is also pre-trained using the image data from COCO so it can be directly used for further tuning.

5.7.5 Results from Initial Testing

After training the model, we tested it on the subsection image from a sample image, as shown in the top position in Figure 5.17. For observation, out of the total results detected by the trained model, 9 segmented objects which correspond to melt-pools have been selected and are shown together in Figure 5.17

Looking back on this approach, the annotation work on the training samples consumes much time and the quality of the drawing of the polygon to indicate the shape of the object can also impact the performance of the training and the resulting model. In fact, as shown in the segmented results in Figure 5.17, some edges do not exactly match the actual shape of the melt-pool and in some case the melt-pool is not significant on the image so it was considered as background rather than as an object. This means that not all melt-pools can be effectively segmented from the image.

We aim to improve on this preliminary result by acquiring more image samples

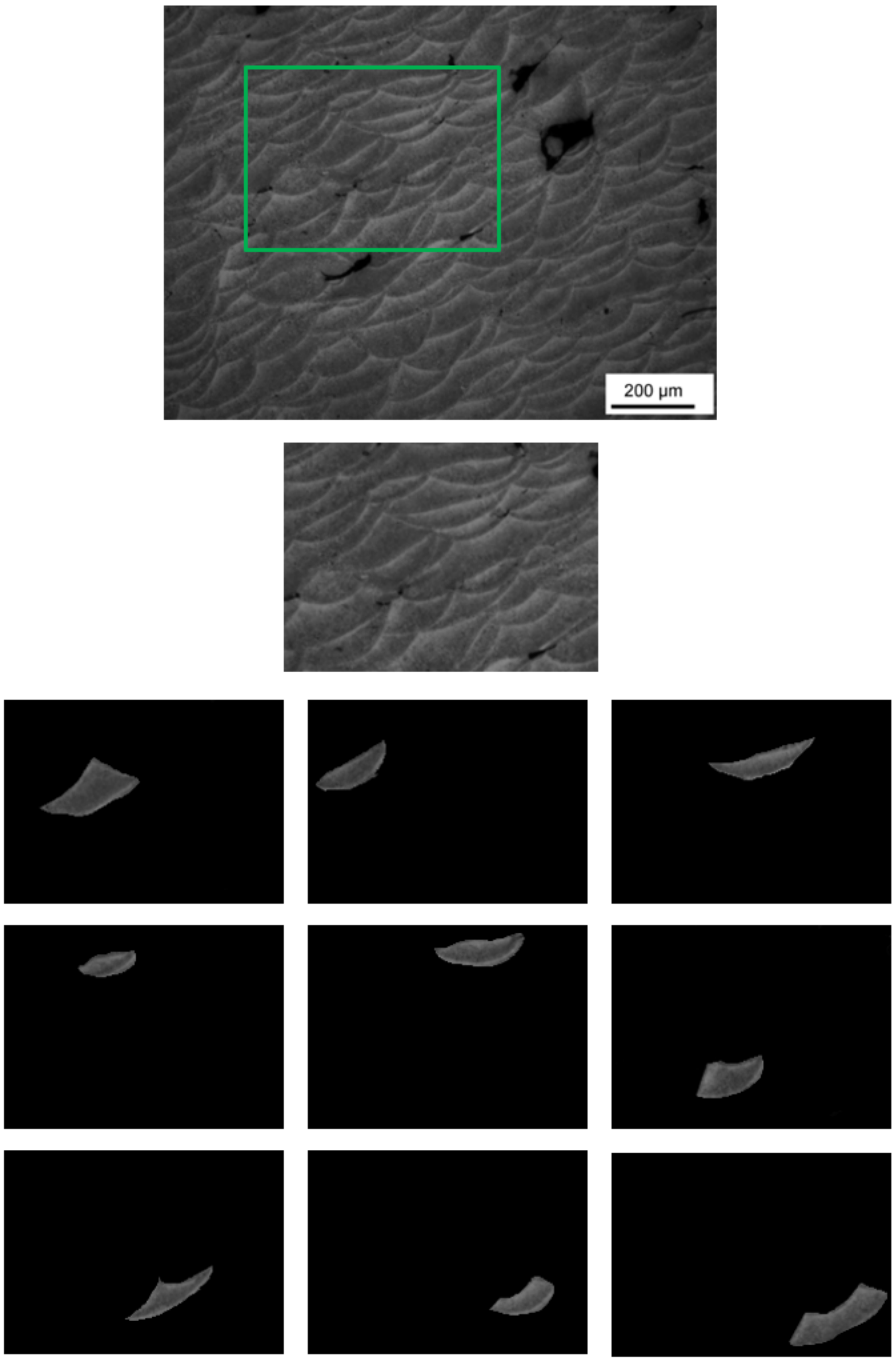

Figure 5.17: Examples of detected melt-pools from the sample image.

with standard quality and to collaborate with experienced engineers and researchers for better annotation work on the images. On the other hand, if possible in the future work, a further investigation into the Mask R-CNN model could also be helpful in the utilisation of such kinds of framework as Mask R-CNN which has various versions and results in terms of its performance on other tasks.

5.8 Conclusions

In this chapter, several studies into object detection related techniques have been presented. We then used some of these techniques as initial investigation to test on image data generated from monitoring of the additive manufacturing process.

The work was somewhat limited by the availability and supply of data and the time taken for annotation and thus this testing on object detection may be brief but we still consider the study and investigation is a good extension into the scope of computer vision-based inspection of data produced from additive manufacturing.

The approaches that have been studied and tested in this chapter show the potential for this form of computer vision to be feasible as assistance for analysis and detection of the detailed structures present in the images generated from monitoring of additive manufacturing. This would include such as automatically counting the number of melt-pools on a window of a layer, the detection and calculation of the approximate sizes of the melt-pools and determining the distribution of the melt-pools or other microstructures. We will leave this as on-going work and a part of the future research.

5.9 Chapter Summary

In this chapter, a deep learning model based on Convolutional Neural Networks (CNNs) has been designed and tested to automatically identify abnormal images in seven types of defect patterns from different image datasets. These datasets include emission images from the AM process of parts made using titanium alloy Ti6A14V along with other patterned images from the DAGM dataset [24]. In the proposed approach that has been illustrated in this chapter, we leverage information about emissions collected through in-situ monitoring and represented as images, one image for each layer deposited onto the object or part being manufactured. These images represent a dataset we have curated and used to train the CNN based deep learning model. The experiments have demonstrated that the model pre-trained and fine-tuned can obtain good performance even with relatively low computational power measured in terms of the number of epochs needed before convergence, and with only a very limited amount of training data.

The model proposed and tested in this chapter can be used as an effective feature extractor and classifier with limited labelled data for training. We believe that the ability to generate more labelled data using the outcome of this deep learning model is necessary, as it not only enables faster convergence (with a limited number of epochs), but it also represents a valuable resource to be used by other researchers. For this reason, there are further investigations into the combination of the model with active learning techniques to develop a framework that can produce good quality labelled data to be used when applying machine learning to AM processes. Thee will be described in the chapters to follow.

Chapter 6

Combining Transfer Learning and Active Learning Feedback in AM Defect Detection

Due to rapid developments in the Additive Manufacturing (AM) industry and the development of monitoring technology used during the AM process, there has been an increase in the size of the data generated, collected and stored during the manufacturing process. As a result, it is common that raw data collected by an AM monitoring system is not labeled and the large amount of unlabeled data usually exceeds the capability of manually analysing and labeling it. This creates an impediment to exploiting this data fully. For example it is challenging to train high performance supervised deep learning models for applications like defect detection with insufficient properly labelled data samples, as is the case here.

As illustrated earlier in Chapter 4, transfer learning is a technique that first performs training on a dataset in one domain called the source domain and then modifies the weights in the model it has learned using data from that source domain and applies the trained model to a target domain that is different from the source. This is done to allow rapid progress in re-training a model into the target domain and to support improved performance of computer vision related tasks such as classification on text [138] and images [139], video recognition [140], hand gesture recognition [141] and human action recognition [142].

On the other hand, active learning [26] is a machine learning technique in which a reduced amount of labelled data is used to train a model but there is interactive labelling of new data points in order to improve the performance of a model trained on data which includes these new labelled data points. As further explanation, active learning is a methodology that prioritises learning the most informative data which potentially has the highest impact on the supervised training of the machine learning model in order to accelerate and improve accuracy of the training process. Both transfer learning and active learning can be considered as design methodologies, and the combination of transfer learning and active learning together allows leveraging small amounts of labelled data to improve the performance of the training process of deep learning models.

This chapter is included in the thesis to describe the proposed method where we apply active learning techniques to a deep learning model developed using transfer learning. This is accompanied by fine-tuning techniques to further investigate the performance of the training process of the model, which also recognises the learner in this active learning process. All this is done in order to further improve the efficiency of data usage in the overall training process and we aim to obtain classification results with higher accuracy and with lower loss for the downstream application, namely the classification of deformities and defects in the addititive manufacturing process.

In addition, this chapter further illustrates the approach of the actively training process of the deep learning model, specifically focusing on the Active Sample selection and Annotation blocks shown in Figure 4.3 in Chapter 4.

6.1 Implementation

This section is a follow-on that continues from the study introduced in section 2.2.1 of Chapter 2. The approaches and experiments described in this section are based on the application of active learning built on top of the transfer learning and a fine-tuning model that was developed in Chapter 4. To develop the deep learning model based active-learner and to construct the active query training iteration pipeline, a set of developing packages for Machine/Deep Learning are selected after investigation for their capability and compatibility. Indeed, though it is a fundamental step, the implementation of the overall active learning architecture is an essential procedure for any of the tests, experiments, evaluations and validations that we present later.

6.1.1 Necessary Software Packages for the Implementation

In comparison with the design process of the overall architecture that has been illustrated in Chapter 3, completing the programminging works for the necessary features required detailed investigation and study into the machine learning packages in Python. There are several Python packages available for the particular construction of the specific parts of the whole processing pipeline we propose to investigate. As a matter of experimental reproducibility, we consider it is necessary to provide a list and an illustration of all the necessary packages used in our study and development of the active learning architecture which we do in this section.

Keras for the Implementation of a Deep Learning Model

Though this Python package has been used earlier in Chapter 4 to carry out all the experimental processes of implementation, transfer learning and fine-tuning, the description of the use of Keras at that earlier point in the thesis was brief, just one single sentence. Thus it is necessary to list this package here in order to give a clear indication of the significance of its use in the thesis.

As an already developed feature in previous illustrations, the learner part of this work, which is based on the CNN deep model, is implemented using the Keras [143] deep learning packages. Keras is a high-level deep learning API of TensorFlow 2 [144] which is an end-to-end open-source machine learning platform with four key abilities:

1. Efficiently executing low-level tensor operations on CPU, GPU, or TPU.
2. Computing the gradient of arbitrary differentiable expressions.
3. Scaling computation to many devices, such as clusters of hundreds of GPUs.
4. Exporting programs to external runtimes such as servers, browsers, mobile and embedded devices.

Based on the advantages of TensorFlow 2, Keras further provides a highly productive interface, essential abstractions and building blocks for developing solutions with high iteration velocity and allows users to take full advantage of the scalability and cross-platform capabilities of TensorFlow.

Software Packages Used for Active Learning

In this section we present work where further study was carried out to select the appropriate software package in order to develop suitable solutions to adapt the work that was developed using Keras, to our active learning pipeline. Initially there were 4 Active Learning Frameworks that had been considered as possible choices for the tasks, these frameworks being modAL [145], ALiPy [146], libact [147] and AlpacaTag [148]. It worth mentioning that PyTorch Active Learning is also an excellent Library for common Active Learning methods. However, as our work is based on the Keras deep learning model, PyTorch was not taken into consideration. Unfortunately, though some of the 4 packages have good functions and features, such as ALiPy and AlpacaTag, they do not have sufficient compatibility to collaborate with our previous work due to dependency and version issues.

The lack of relevant documentation and worked and illustrative examples for learning was also a major concern in using these packages as part of the implementation of our work. For the remaining two Active Learning Python packages modAL and libact, they have a similar interface and are both based on scikit-learn [149], which is one of the most popular libraries for general machine learning in Python. However, in comparison, the documentation for modAL has better detail and worked examples which provides more support in learning and practice for coding work. More importantly, modAL can seamlessly integrate Keras models into the active learning workflow by adapting the Keras deep learning model using wrappers API. This feature further makes modAL the most suitable choice from the overall listed active learning frameworks to fulfill our implementation requirements of an Active Learning mechanism in our research.

6.1.2 The Active Learning Mechanism

As a brief summary of the concept behind our implementation, the main model of the active learner is based on the outcome of the classifier mentioned in Chapter 4, which has been implemented using Keras with the TensorFlow deep learning platform in Python as the back-end. To build the whole Active Learning mechanism, another Python package for Active Learning methods named modAL was used in combination with Keras. As further illustration, We started this work from the standard Keras encoded VGG-16 convolutional neural network model, through the scikit-learn wrappers API to adapt this Keras model into an Active Learning pipeline that was constructed using modAL which is also based on the scikit-learn framework. In this way, we were not only able to achieve the goal of an implementation for the designed Active Learning architecture, but also were able to leverage the power of the scikit-learn library for further tasks like model evaluation and optimisation. The concept underlying our implementation is shown in Figure 6.1.

The Algorithm for the AL mechanism

This section presents the logic and steps of the Active Learning process in the implemented architecture and summarises them into the form of an algorithm. The designed overall process is based on a pool-based sampling scenario where we apply an active learner onto the emission dataset. In this scenario we assume a small set of labeled data L and a large set of unlabeled data U. Normally, it is also assumed that $|L| << |U|$, but as our current dataset is relatively small, the initial labeled

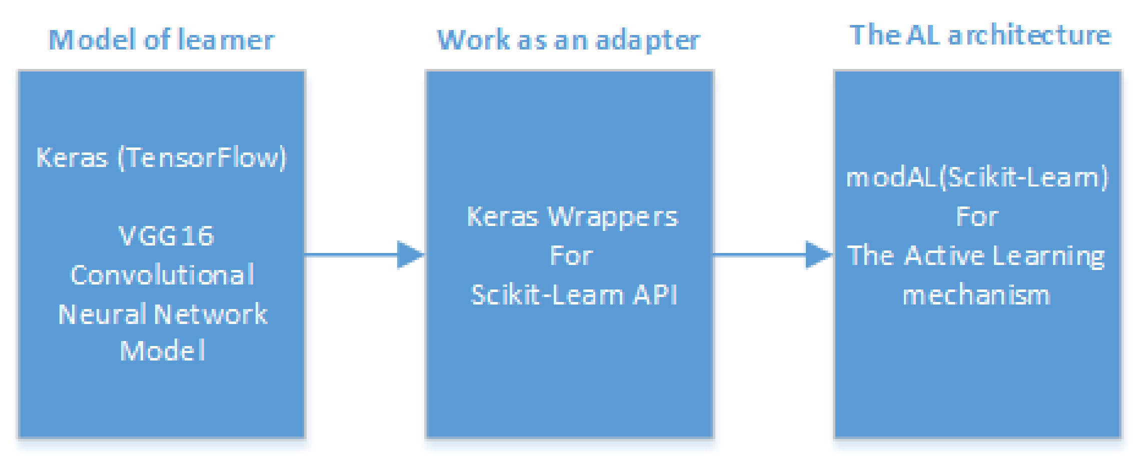

Figure 6.1: Concept underlying the implementation of the Active Learning architecture

data L should be even smaller than U, though it may not be too much smaller than U in some situations for experimental purpose. L is considered to be the initial training set for the learner while U is specified as the pool and assumed to be an unlabeled dataset.

In the following steps, after the initial training using L, the active learner starts to draw samples from the pool U in order to query the unlabeled dataset for the most informative training samples according to the assigned query strategy setting. Then the queried samples are sent for labeling to the oracle. In real world tasks, the oracle should answer the query and assign labels to the samples, but for the experiments the label of the samples in the pool U are already known as the ground truth. The implemented learning mechanism will simulate a human annotator and automatically label the queried samples for the learner, then the newly labeled samples will be removed from the pool and added to the training dataset. As a consequence, the training dataset will be enlarged in this manner to provide improved informativeness in the training of the active learner. Thus, the performance the learner model will also be continuously adjusted by applying fine-tuning using the updated training dataset in each query iteration until the stopping criteria are satisfied.

For a more detailed illustration, a pseudo-code representation of this Active Learning mechanism is given as Algorithm 4. In the procedure, there are variables that can be tested to investigate their related impacts on the performance of the overall learning process of the Active Learning model, and these include the initial data for training X_{init}, the number of Queries Q_i and the number of queried samples N in each Query.

Algorithm 4 Active learning with deep learning model

```
 1: procedure AL FOR EMISSION IMAGES ()
 2:     Given ($X_{train}$, $y_{trian}$) = ($X_l$,$y_l$)
 3:     Assemble initial data for training ($X_{init}$, $y_{init}$); ($X_{init}$, $y_{init}$) $\in$ ($X_{train}$, $y_{trian}$)
 4:     Create the pool by removing the initial data from the training dataset $X_{pool}$
    $\leftarrow X_{train}$ - $X_{init}$
 5:     Current dataset used to train the initial classifier $X_{teach}$ = $X_{init}$
 6:     Train initial classifier $C_{init}$ using $X_{teach}$
 7:     for all queries $Q_i$ do
 8:         Query N samples to assemble $X_{query}$ from $X_{pool}$
 9:         Enlarge the dataset for training $X_{teach} \leftarrow X_{teach} + X_{query}$
10:         Teach the classifier using newly obtained $X_{teach}$
11:         Remove queried instances from pool $X_{pool} \leftarrow X_{pool}$ - $X_{query}$
12:     end for
13: end procedure
```

6.2 Experiments

This section involves the description of the experiments carried out on applying the implemented Active Learning mechanism on the emissions dataset we collected. The aim of the experiments is to investigate the impact of different variables on the performance of the training during the process of actively querying, labeling and learning through the iterations. The tested variables in the experiments are the initial data for training X_{init}, the number of queries Q_i and the number of queried samples N in each Query. The aim of the experiments are focused on research into effectively utilising the available data samples with a limited querying budget and trying to enhance the performance of the training under such an experimental setup.

6.2.1 Arrangement of the Dataset, Parameters and Settings

Arrangement of the Dataset for Experiments

For this Active Learning section, the data used is focused on the emissions image dataset. The dataset for the experiments is based on the same dataset of Class 7 patterns (refer to Section 5.1 earlier) that have been used for the earlier experiments when applying transfer learning and fine-tuning techniques on the deep learning model developed in Chapter 4. Originally, the sample images which represent the melt-pool conditions of the printed layers in a group of dogbone shaped testing parts of Titanium alloy (Ti6Al4V) during the AM process, were manually inspected in order to select 150 images as defect samples and another 150 images as normal samples. This was done to form a balanced dataset.

In the setup of this experimental section, 240 samples which is 80% out of the total 300 labelled samples are used for testing and the remaining 20% are reserved for further use in later sections. The querying budget for the number of samples that will be requested for labeling will vary according to the different criteria in each experiment where testing scenarios will include different initial conditions, training lengths, number of queried samples and the total number of queries to have been executed. All this is done to investigate the impacts of these factors on the performance of the training of the model.

Parameters and Settings

This section presents the values of the general parameters used in the overall architecture for the Active Learning based training and labeling process. The Python packages for the implementation have already been listed earlier in Section 6.1.1. In addition, the experiments are executed using Google colab [150] for online GPU support. Table 6.1 contains a listing of the hyperparameters and query strategies used for the training of the active learner.

Stopping criterion: The main purpose of these experiments is to test the performance based on the number of total samples queried. The stopping criterion

Table 6.1: Hyperparameters used for training the deep learning model.

Name	Type / Value	Description
Optimiser	Stochastic Gradient Descent (SGD)	Optimisers are used to change the attributes of the neural network to reduce the losses.
Loss function	Binary cross entropy	The loss function computes the quantity that a model should seek to minimise during training.
Learning rate	0.001	The step size at each iteration while moving toward a minimum of a loss function during the training.
Query strategy	Uncertainty sampling	The strategy used to select the most informative samples to be queried from the pool.
Evaluation metric	Accuracy, Loss	The metric is a function to judge the performance of the model.

emphasised in the query budget is that when reaching a certain number of queried samples from the pool, iterating of the Active Learning will cease.

Training length: For the training length in each query iteration, to efficiently use computational power, the training length is set at 25 epochs for the reason that according to Figure 5.8 the performance of the classification model shows the most significant improvement before the training of epoch 25. For that same reason, the training length of each query iteration of the Active Learning process is also set to 25 epochs.

6.2.2 Experiment A: Learning with Different Numbers of Samples and Queries

To start the first experiment, as the initial learner, the CNN-based classifier with the transferred weights from ImageNet as mentioned in the previous section, was trained using 100 prepared samples for initial fine-tuning. 100 samples were then removed from the pool as they had already been used in the training and fine-tuning dataset.

This test aims to investigate the impact of the number of queried samples in each query when the same number of total training samples is available. The query strategy used in this test is the Least Confidence query strategy. The test was performed with 2 different combinations. In test 1, 5 samples in each query for 20 queries and in test 2, 20 samples in each query for 5 queries. In this way a total of 100 samples are involved in the query/learning process. The result of the classification accuracy during the Active Learning process of the 2 different setups are shown in Figure 6.2.

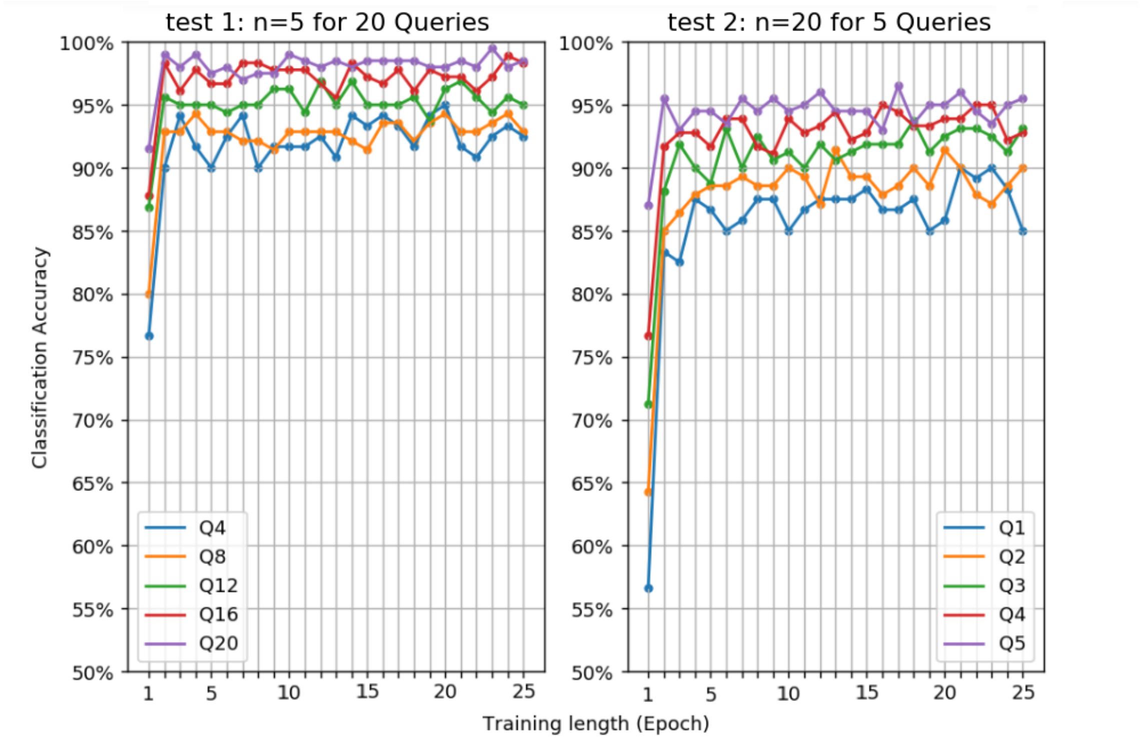

Figure 6.2: Accuracy of testing on the emission image dataset, using a model with pooling and fine tuning.

For comparison, in the left chart of Figure 6.2 the lines of the queries Q4 Q8 Q12 Q16 Q20, which are 5 queries out of the total of 20 queries, are selectively shown in order to match the condition of the same total number of samples used in the corresponding 5 queries in the right chart, which are labeled as Q1 to Q5. The loss during the Active Learning process of the 2 different setups is shown with a similar selection in Figure 6.3.

The results show that the first test is generally higher in accuracy when reaching

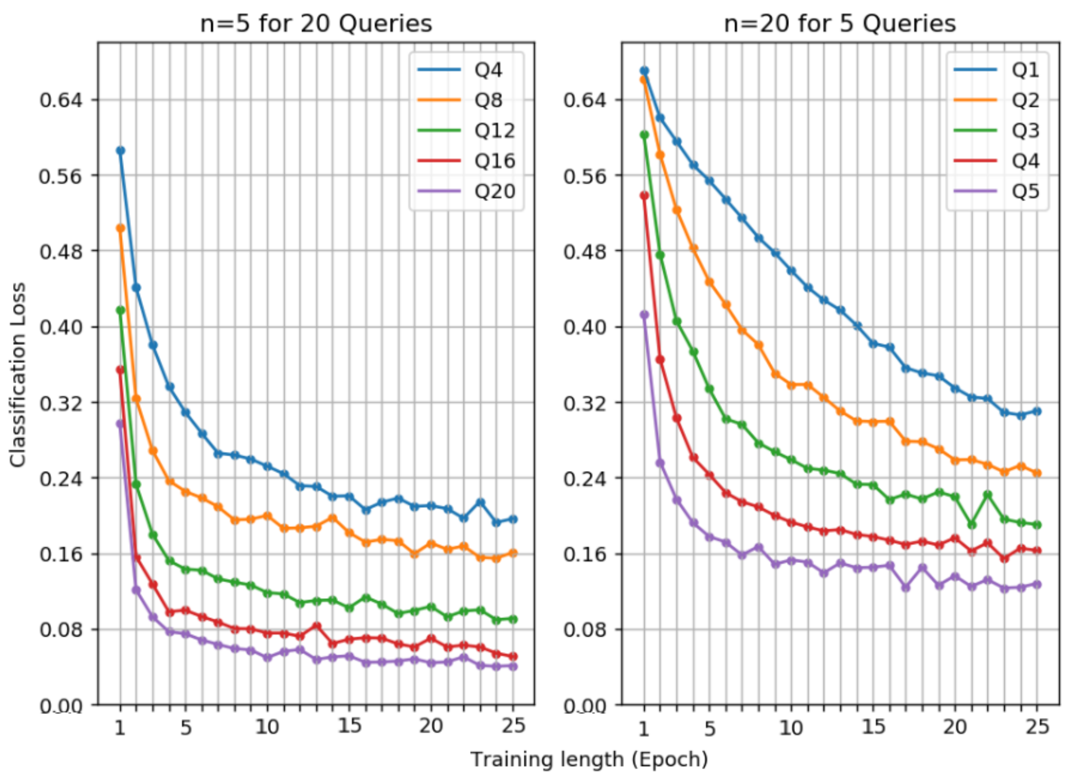

Figure 6.3: Loss of testing on the emission image dataset, using a model with pooling and fine tuning.

the relatively steady state in each query iteration. When a total of 100 more samples are used in the both tests, the first test shows around 97% and with less fluctuation in the overall training accuracy while the value in the second test fluctuates at around 95%.

For the accuracy at the first epoch in each query iteration, test 1 also shows higher values that start at over 75% and reach about 92%. In comparison, test 2 starts as low as 55% for Q1 and reaches about 87%. It is predictable that the accuracy value of Q4 in test 1 is considerably higher that Q1 in test 2 for the reason that the classifier in test 1 is more experienced as it has received training in the previous queries while in test 2 the training will have just begun. However, when further examining with more training epochs, when the accuracy no longer improves with more epochs in the current query iteration, where the learner reaches the limit of the learning progress with the same total amount of training samples for example 20 additional samples both in test1 and test 2, test 1 still shows a higher level of accuracy in general.

From the view of the loss in the Active Learning process, test1 with fewer samples in each query for more queries than the corresponding setup of test 2 results in earlier convergence at a lower value of loss, which means higher confidence in each classification. This also agrees with the fact that there are less fluctuations in the accuracy at later stages of test1. This behavior is similar to human learning activity: one can compete a learning task with a shorter time when the learning content is less and gain a better understanding after certain rounds of reviewing, until final understanding. There will be further illustration of this in the discussion section.

6.2.3 Experiment B: Impact of Initial Training with Different Numbers of Samples

In this experiment, we investigate the impact on the overall performance of the learner when changing the number of samples used for the initial training and the number of samples used in the Active Learning. Three sets of tests are reported where the number of samples for the initial training were set as 20, 60 and 100 respectively. Thus, these 3 experimental setups are referred to as ini20, ini60 and ini100 in the following content of this section. We keepi the condition the same as in the last experiment that 5 samples were asked in each query until a total of 200 samples were processed for all three tests. The results for each test at different total numbers of samples used in the queries, which are 120, 140, 160, 180 and 200 corresponding to line colors blue, yellow, green, red and purple with relevant query number labelled, in Figure 6.4.

Overall, the results in Figure 6.4 show that a greater number of active-selected samples at an early stage of the training can further improve the the final accuracy in the training with relatively lower fluctuation in classifications results. The boxplot of the very last query in the learning process for each of the three tests are shown in Figure 6.5, where outliers are reducing in number as well as the distances from the outliers to the caps of the corresponding box are also reducing while the active selected samples is increasing.

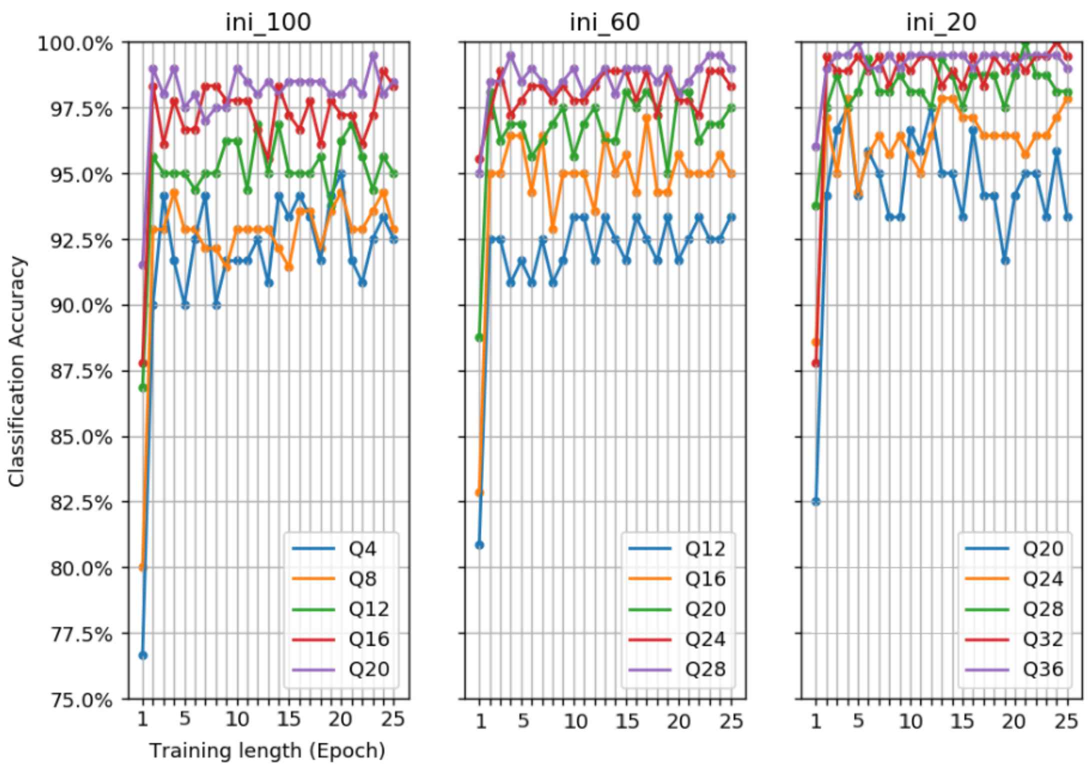

Figure 6.4: Classification accuracy of corresponding numbers of samples queried in the learning process with different initial sample sizes.

The mean values for accuracy and the values of standard deviation for each of the three tests are shown in Table 6.2. The values were calculated after removing the outliers.

Table 6.2: Average accuracy and standard deviations in the very last query iterations for each of the three tests.

	Number of initial samples		
	100	60	20
Average accuracy	98.25%	98.73%	99.38%
Standard deviation	0.0055	0.0045	0.0026

According to Table 6.2, when a greater proportion of training samples are active-selected in the tests using 20 initial samples (thus, 80 further samples are actively learned for the first 100 training samples), compared to the test with 100 initial random samples, although the improvement in average accuracy may not be significant at around 1% which is from 98.25% to 99.38%, the standard deviation is

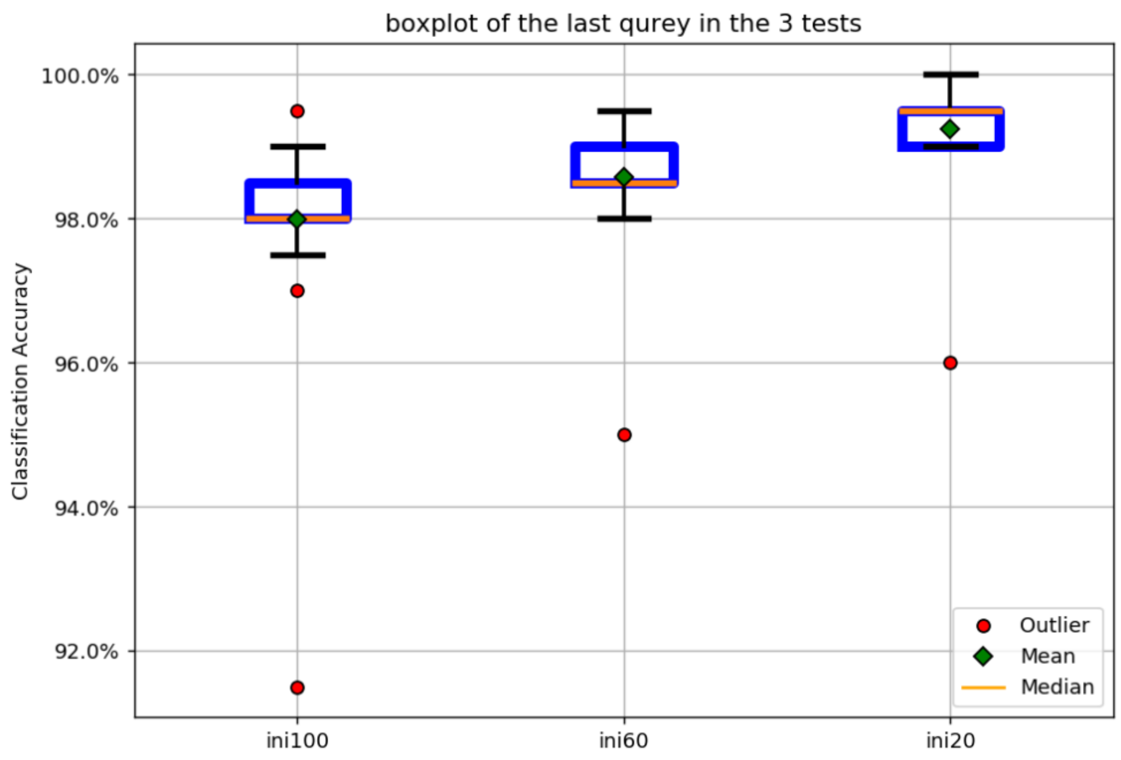

Figure 6.5: Boxplots of the very last query in the learning process with different initial samples for each of the three tests.

smaller showing that overall classification accuracy is more stable if Active Learning is applied earlier.

If we further observe the performance of the training accuracy of the very last query of the three experimental setup in detail, the plotted chart in Figure 6.6 shows that the greatest difference in the values of the training accuracy during the last query happens at the first epoch, where the classification accuracy is relatively lower than the mean value of the training accuracy from the 2nd epoch to the 25th but this difference decreases as the total number of active learned samples are increased in each of the experimental setups.

According to this particular result, two points can be considered for discussion:

1. When the total available training dataset is small, over-fitting can happen. In the situation of over-fitting, the model tries to fit every data point in the training set. Because the learning material available may not be sufficient for the model to fully understand the general features of the being classified object, it tends to “memorise” rather than “understand”. Thus, the model

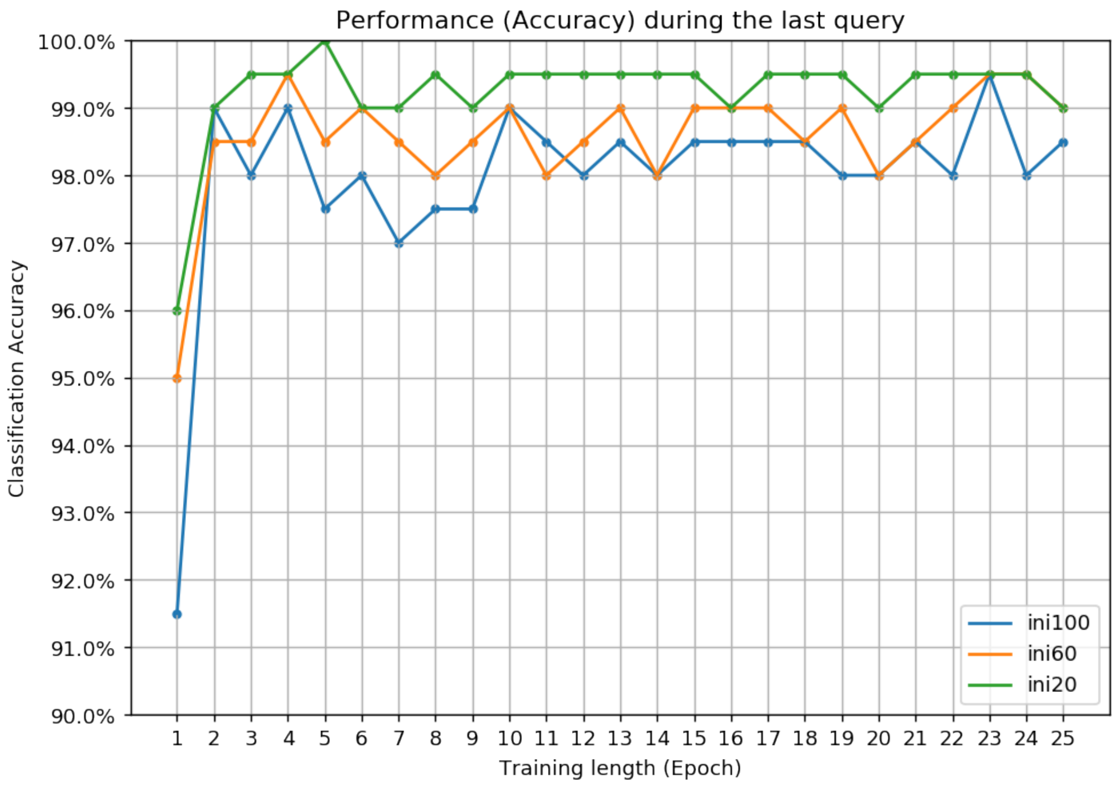

Figure 6.6: Performance of training accuracy of three experimental setups with increasing epochs in the last query iteration.

can be inaccurate for new data samples and vulnerable to noises or outliers. This point can be an explanation for the performance of training accuracy as it drops in the first few epochs when the new query iteration starts. However, as the total size of the training dataset increased by active queries and labeling, the total number of labeled samples in the training dataset is approaching the level to be sufficient for the model to understand the data. This is also the reason for the difference between the values of the training accuracy at the first epoch and the average value after fine-tuning getting smaller as the query iterations proceeding. Fortunately, in Section 5.3.3 we have demonstrated that the size of our emission images dataset is large enough to avoid the over-fitting problem and complete the training of the CNN-based model in order to achieve a relatively high classification accuracy, both in the training and validation process even before involving the Active Learning techniques. When over-fitting happens during the training, the validation accuracy will drop and the validation loss will increase as the training continues, but this is not the case in Section 5.3.3. So even though there can be some level of over-fitting occurring during the active labeling and dataset enlarging process, the final results of the training should not be significantly affected by over-fitting.

2. Based on the performance found in the experiments and in addition to the current stopping criterion, which is the querying budget, when applying this Active Learning framework to the labelling task, the stopping criterion can be further extended to consider the factors of the accuracy level at the first epoch in the query iteration. For example, when accuracy reaches a certain stable value, the active queries will be stopped, then the model will start to classify the rest of the unlabeled samples and assign labels from the classification results. There will be further illustration on the labeling and evaluation work in Chapter 7, which is the next chapter. Further discussion on the overall performance of the experiments is in the next section.

For the aspect of training loss, the performance is also similar to what has been shown for the training accuracy that an increased proportion of active selection and querying on the data samples yields lower loss. In Figure 6.7 the results for the values of the training loss in each test (ini100, ini60 and ini20) at different total numbers of samples used in the queries, which are 120, 140, 160, 180 and 200 corresponding to the lines colored blue, yellow, green, red and purple with the relevant query number labelled as well as Figure 6.8 which compares the change in value of the loss as the training epoch increases for the 3 different testing setups.

6.2.4 Further Discussion of the Experiments

In real world applications using the Active Learning process for AM emission images, queries need to be confirmed by a human annotator. The workload for the human annotator is thus highly depend on the volume of the total number of samples queried. In experiment A, where we used learning with different numbers of

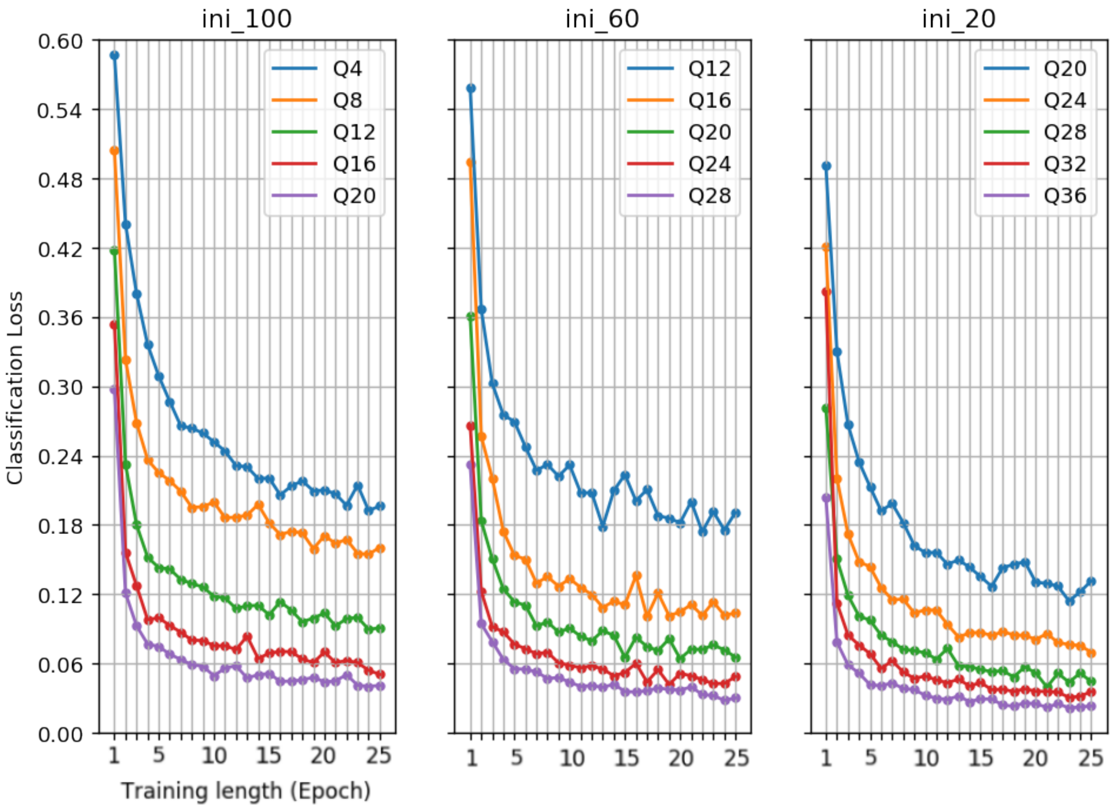

Figure 6.7: Training loss of corresponding number of samples queried in the learning process with different initial sample sizes.

samples and queries as described in Section 6.2.2 by maintaining the same number for the total actively queried samples, the workload for the oracle remains the same. The results from experiment A with different test setups showed that the method using fewer samples in each query but with more queries, performed better in the classification tasks. In fact, according to Figure 6.2, test 1 out-performed test 2 in experiment A by reaching about 95% in accuracy using an additional 60 queried samples compared to the 100 queried samples used in test 2. This means that the human annotators are able to obtain the similar performance for classification with 40 less samples identified and thus the approach can reduce the human workload in the whole Active Learning process.

Discussion on training performance concerning accuracy output

There may be concerns with the results shown in Figure 6.2 namely that for the performance degradation observed between active learning iterations, it is important

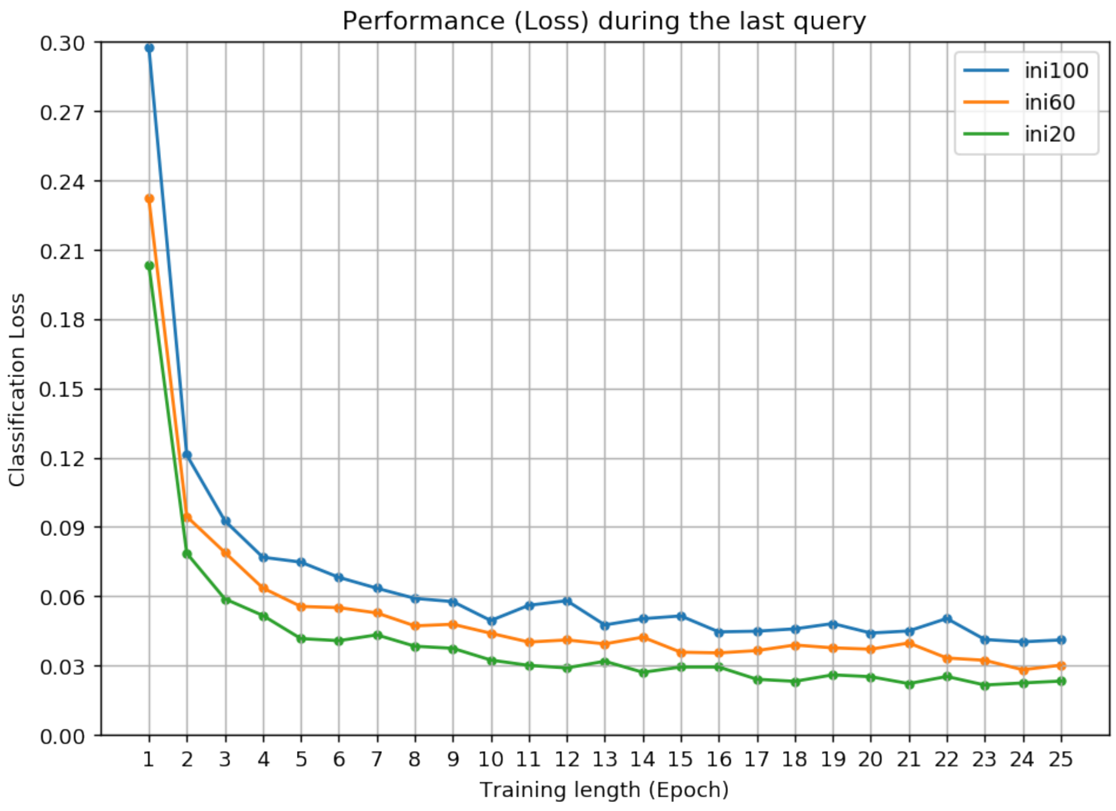

Figure 6.8: The performance of training loss of the three experimental setups with the increasing epochs in the last query iteration

to clarify that this experiment focuses on the training process. The recorded classification accuracy represents the performance on the current training data (training accuracy), rather than the validation dataset. As the training data are updated with each active selection (or query) iteration, the composition of the training data changes dynamically. Consequently, at the beginning of each iteration, there is a portion of new data unseen by the classifier. At this stage, the classifier only retains the weights from the last iteration and is not updated by even a single epoch. This condition, which we name it as "stage 0", is when the first round of classification occurs using the non-updated classifier on the updated training data. These initial results are recorded as the first point on the plotted line for each presented query in Figure 6.2 resulting in the decreased accuracy between active learning iterations under the "stage 0" condition. It is noteworthy that after the first round of training, the training classification accuracy quickly rises in the second epoch of each query iteration, where the classifier has been trained for one epoch and learned from the updated training data.

Furthermore, in the early stages of the experiment, the classifier is trained using an insufficient number of training samples. Therefore, it is not surprising that the classification results in early queries exhibit relatively low accuracy on unseen data newly selected from the pool at “stage 0”. As training progresses, the drop at the beginning of each query iteration becomes less significant. This is because the classifier has been progressively trained and becomes more experienced with the overall dataset. Consequently, it can perform classification with higher accuracy even under the “stage 0” condition.

Regarding the training duration of each query iteration, we consider it appropriate because the relevant loss values shown in Figure 6.3 reach convergence after the training of each query iteration. Longer training may lead to overfitting, so we will not consider increasing the training duration in each query iteration.

Discussion on the training performance in relation to the sampling size in each query

For further discussion on the relationship between the sampling size in each query and the associated overall training performance, it is commonly believed that a larger sampling size in each query will result in better training results. This is because with a larger sample size, there is a higher chance of capturing diverse instances from the dataset. This diversity can aid in building a more representative training set, covering a wider range of scenarios and variations present in the data. However, it is essential to note that the effectiveness of larger sample sizes in active learning can vary depending on the specific characteristics of the dataset, the complexity of the task, and the behavior of the learning algorithm. Conversely, smaller, more focused queries may prove more effective, particularly when targeting specific areas of uncertainty or high model confidence. Therefore, experimenting with different batch sizes and sampling strategies is crucial to determine suitable approach for a particular task and dataset. This necessity drove the investigations conducted in Section 6.2.2

and we consider that it is normal to obtain observations that are surprising and counter-intuitive in such investigations

Furthermore, it is commonly considered that labeling noise or errors may occur when annotating individual samples. By selecting a larger batch of samples at once, any labeling noise in individual samples may be diluted or balanced out by the overall consensus of the batch, potentially leading to more accurate annotations. However, this was not considered in our experiments as all labels used were correctly labeled and served as ground truth, thus avoiding noise interference in the main focus of the tests.

Further investigation into the relationship between the query sampling method and training performance could be an interesting topic and may be considered as a direction for future work.

6.2.5 Discussion on Experiment B

For further investigation, the result of experiment B where we looked at training with different numbers of samples shows that applying Active Learning at the early stage of the training will improve the learning of the deep model and will result in higher final accuracy and confidence of the classification tasks. One clarification to make is that in this experiment, we used a seed because we only wanted to keep the number of initially selected samples as the variable and maintain consistency for other factors and conditions. However, this was only for the investigation stage where the variability of the initially selected number exists. In further applications of the active training method, we will actively select all the samples, thus no initial sample will be needed, and consequently, the seed is no longer necessary in that situation. In future development, when this training approaches a high enough level of performance (for example, over 95% in classification accuracy), the classification and training processes can be fully automated. In that case, the Active Learning mechanism can be used to label new data without the help of a human oracle and can use the newly obtained data to reinforce the training dataset.

6.3 Chapter Summary

In this chapter we designed and tested a deep learning model based on Convolutional Neural Networks, transfer learning and Active Learning techniques to automatically identify abnormal emission patterns in an image dataset we created from the AM process of titanium alloy Ti6A14V. In our approach, we leverage the Active Learning technique to further improve the performance of the training of the classification model while trying not to increase the potential human workload required in the process by limiting the total number of samples queried.

The experiments in this chapter have demonstrated that the model with transfer learning and Active Learning can obtain good performances in the training process even with relatively low computational power and a limited number of labelled data samples. This approach can be further developed to aid in automatic labelling of emission image data. We believe the ability to generate more labelled data using the outcome of our model is necessary, as it not only enables faster convergence (with a limited number of epochs), but it also represents a valuable resource to be used by other researchers.

The next chapter will discuss in more detail the labelling framework and related works for evaluation.

Chapter 7

Closing the Loop: Evaluation of the Overall Architecture

In the previous chapter, by using the combination of transfer learning and fine-tuning with active learning, we developed a framework that involves a human-in-the-loop approach to continuously improve the training process of a deep learning model for accurate classification. However, while all the experiments focus on the training process and the results are mainly shown in terms of training accuracy and training loss, the validation is also an important step for such an active learning framework. This chapter aims to demonstrate the validation work of the developed framework. It begins with evaluation on performance using several common metrics and discusses results, then follows on with tests performed using different datasets including the DAGM patterns for further evaluation. After validation and evaluation using the available datasets, we will also discuss on the stopping criteria that is related to the active learning approach and discuss our view on what should be the point where the classifier can execute auto-labeling as a result of a classification outcome.

In addition, this chapter further illustrates the approach used to close the loop of the labeling mechanism, specifically focusing on the auto-labelling block shown earlier in Figure 4.3 in Chapter 4. Furthermore, this chapter proposes an approach to address the third research question (RQ3), which involves completing the feedback loop with human-in-the-loop (HITL) features. After evaluating the classifier, the labeling mechanism continuously improves and enlarges the training dataset by adding more accurately labeled samples. Theoretically, the enlarged dataset offers several advantages. Firstly, with a larger and more diverse dataset, machine learning models can better capture the complex relationships between process parameters, material properties, and resulting part quality in additive manufacturing. This leads to improved generalisation performance, where the model can accurately predict part quality for new and unseen combinations of parameters. Secondly, as the training dataset grows, the model becomes more adept at identifying subtle patterns and anomalies indicative of defects in manufactured parts. This increased sensitivity improves the model's ability to detect defects early in the manufacturing process, reducing waste and improving overall quality control. Thirdly, additive manufacturing processes exhibit inherent variability due to factors such as material variations, machine settings, and environmental conditions. By training on a continually enlarged dataset that encompasses this variability, machine learning models can learn to adapt and account for these variations, resulting in more robust predictions and fewer false positives or negatives. Overall, continually enlarging the training dataset in machine learning applications for additive manufacturing enables more accurate predictions, improved defect detection, better adaptation to variability leading to improved performance and generalisation to unseen data. Consequently, these improvements can enhance performance metrics such as classification accuracy in additive manufacturing defect detection. While a practical case study applying the proposed approach will be presented in Chapter 8, Chapter 7 focuses on presenting the related method.

7.1 Evaluation Based on Metrics from Validations

In this section we present work where a set of validation tests were carried out for the evaluation of our framework. The tests involve balanced and unbalanced datasets of emission images and all the 6 datasets from the DAGM patterns in the three subsections. The metrics used in the tests include the Receiver Operating Characteristic (ROC) curves and the corresponding Area Under the Curve (AUC), the confusion matrix and the calculated precision, recall, f1-score as well as accuracy. These metrics are commonly used to evaluate the performances of classifiers in machine learning.

7.1.1 Metrics for Tests Using a Balanced Emissions Dataset

This subsection investigates and evaluates the performance of the developed architecture based on a CNN backbone with transfer learning and fine-tuning and with active sample selection approaches. In the first test, we used a balanced emission dataset and execute a set of validations at the end of each query iteration during the active learning process. The metrics collected through the series of validation in the queries are:

1. The Receiver Operating Characteristic (ROC) curves and the corresponding Area Under the Curve (AUC) of ROC;
2. The overall accuracy of classification from the validation in each active query iteration;
3. The confusion matrix that indicates the number of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) as well as the classification figures for precision, recall, f1-score, which will also be recorded as part of this test in a table.

The experiments in this section are an extension from previous experiments in Chapter 6, where the total sampling budget is 200 for the active queries and the training. Out of these, 10% of the samples (therefore, 20 samples) are used for initial tuning to give a basic weight modification for transfer learning after starting from pre-trained weights from ImageNet data. The remaining training samples are separated into 36 queries with 5 actively selected samples by the learner and where the system asks the annotator for the label of the selected samples. After that the learner will be re-trained (fine-tuned) with the newly updated labeled data points generated in the current iteration and the weights of the deep representation for the classifier is consequently updated. The validation dataset is then used for testing this updated classifier and relevant metrics are collected to evaluate the performance. The validation dataset consists of 30 positive (defect) and 30 negative (normal) samples to create a balanced dataset for the reason that if the dataset is highly unbalanced the meaning of the accuracy score will be significantly reduced. The samples in the validation dataset have never been seen by the learner.

ROC and AUC

To evaluate the performance of the architecture and the training approach, Figure 7.1 shows the ROC curves by plotting the true positives rate (TPR) against false positives rate (FPR) and the corresponding AUC is depicted in the chart. Note that the ROC remains unchanged after Query 13 as the model has reached a point where even with minor fluctuations in the validation accuracy, the change in TPR and FPR is not significant to cause the shape to change in the ROC curve from query 13 to query 36 which is the final query in the active learning cycle. Thus, 6 plots are selectively shown out of the ROC curves from the first 13 queries, only when the ROC curves have observable changes or improvements.

It is noticeable that although the AUC is already very high in value on the first few query-training cycles, the accuracy score for the classification is relatively low at the beginning as the training is not yet sufficient. This is also due to the fact that in this classification task, we are transforming a continuous model output into a binary prediction. The accuracy score is calculated using the number of correctly classified samples divided by the total number of classified samples. The result of a classification task not only depends on the distribution of the probability of each prediction, but it also depends on the classification threshold, according to which the decisions are make for the judgement of a class. On the other hand, ROC is based on the probability of prediction and can be calculated without the specific classification threshold. Thus, the performance of the ROC curve and the accuracy

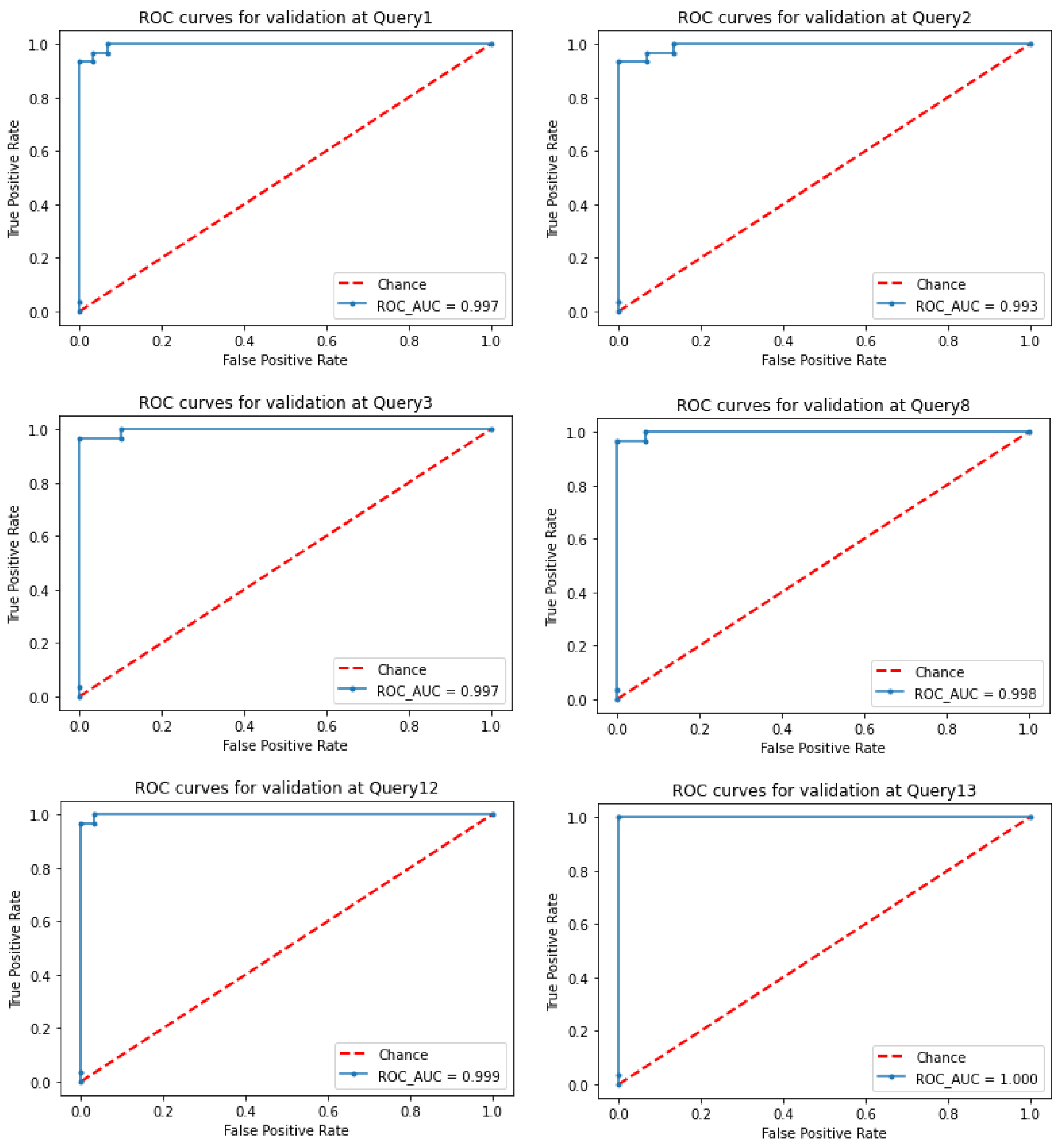

Figure 7.1: Changes in ROC and AUC in the first 13 queries.

score do no contradict one other. In fact, they both improve through the training process. However, the AUC with a value of 1 on later stages of the training, such as the AUC after query 12 in this test, does not mean the model is “perfect”. It only stands for the ability of the current classifier to discriminate between the positive and negative classes in the current validation dataset. The ROC curves and related AUC in the charts indicate that the classifier is well capable for the task after about 13 queries.

Accuracy Scores

Accuracy is another important metric to evaluate the performance of the framework. To observe the overall trend of change in the accuracy score of the classification tasks through the validation process in each query iteration, all the relevant accuracy scores are recorded at the end of the validation test and plotted against the corresponding number of queries. We refer the reader to Figure 7.2.

From the overall performance of the accuracy score we find that the classification accuracy of the model increased rapidly during the first few queries. At around the 13th query the model is able to correctly identify all the samples in the validation dataset with a relatively stable performance after that point in the training process. This is also the point around which the value of the AUC reaches 1.000 in the overall validation. Due to the fact that this balanced validation dataset is small and the total number of samples is only 60, resulting in the fact that even a single classification can change the percentage of the accuracy score by approximately 1.7%. There will be a further test using a bigger validation dataset in Section 7.1.2.

Confusion Matrix and Classification Report

For further detail, the confusion matrix with the relevant numbers for TP, TN, FP, FN as well as the precision, recall and F1 score are shown in Table 7.1. The overall test is 36 queries to go through the 200 labeling budget, but here we only listed the first 20 queries due to the fact that convergence happened at around Q12, after which the performances does not show significant fluctuation and improvements. At around Q20, the classification is correct on all the samples.

In the metrics from the classification report, precision is a good measure to determine when the cost of false positive is high which could be the case, for example, for email spam detection. However, in defect detection for additive manufacturing, a false positive means a defect is detected when there is no defect, so it may cause the engineers to check on the situation but does not demand further action if it turns out as a false positive.

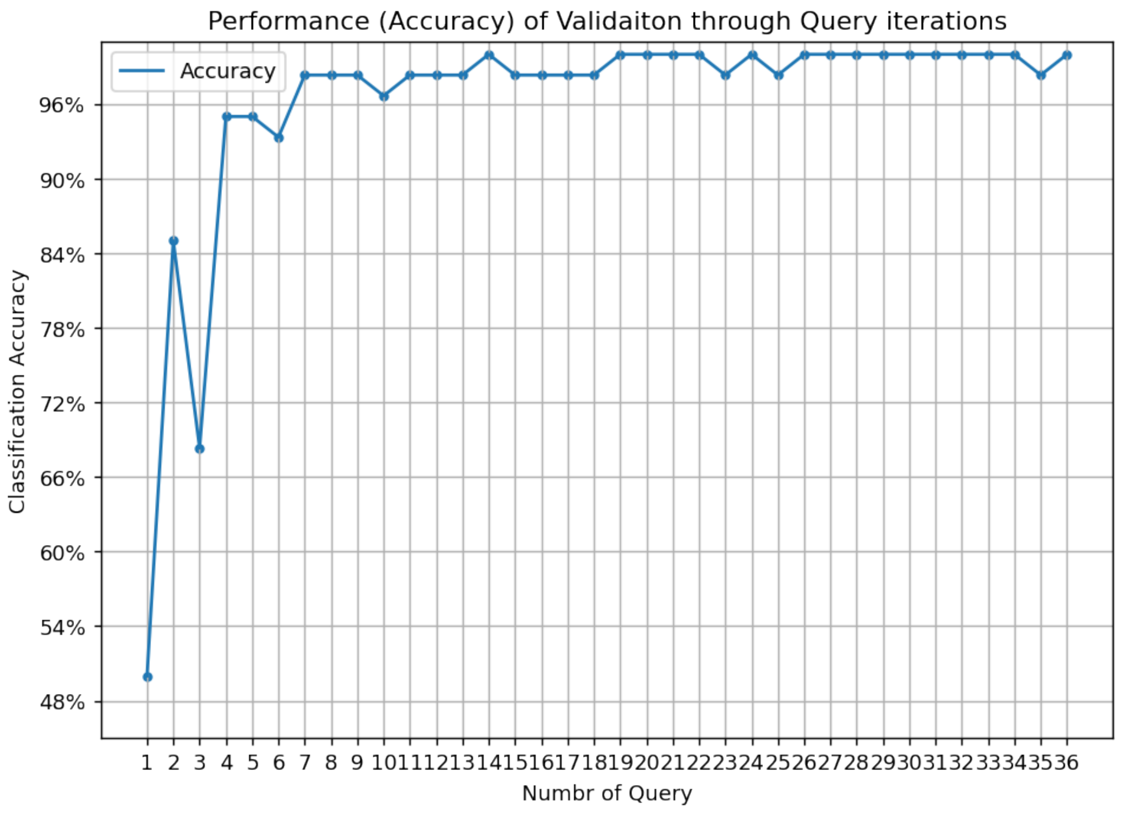

Figure 7.2: Accuracy scores of classification tasks through the validation process for each query.

In turn, Recall is calculated as the ratio of true positive to the total actual positive, which is the sum of true positives and false negatives. Thus, it is a suitable model metric when there is a high cost associated with false negative, such as undetected defects during the manufacturing process which may result in failure in the whole build of the part. For labeling tasks, we would prefer all labels from different classes to be correct. In this situation, F1 score can be a good metric to be considered as the calculation of F1 seeks a balance between Precision and Recall and it also works well when the number of negative samples are large compared to the number of positive samples.

Accuracy can be misleading when the dataset is highly unbalanced and the classifier tends to guess on the majority class. In our research, as the current training and validation data are both balanced, and the overall accuracy for both classification and validation are relatively high, using accuracy as an evaluation metric may not result in significant error, but in further research, we may focus on a more suitable metric especially when the dataset is highly unbalanced. In the next section, the validation will be based on an unbalanced dataset.

Table 7.1: Confusion matrix and classification report for Q1 to Q20.

		TN	FN	classification report			
Query	Label	FP	TP	precision	recall	f1-score	support
Q1	negative	30	0	0.50	1.00	0.67	30
	positive	30	0	0.00	0.00	0.00	30
Q2	negative	21	9	1.00	0.70	0.82	30
	positive	0	30	0.77	1.00	0.87	30
Q3	negative	30	0	0.61	1.00	0.76	30
	positive	19	11	1.00	0.37	0.54	30
Q4	negative	28	2	0.97	0.93	0.95	30
	positive	1	29	0.94	0.97	0.95	30
Q5	negative	28	2	0.97	0.93	0.95	30
	positive	1	29	0.94	0.97	0.95	30
Q6	negative	27	3	0.96	0.90	0.93	30
	positive	1	29	0.91	0.97	0.94	30
Q7	negative	30	0	0.97	1.00	0.98	30
	positive	1	29	1.00	0.97	0.98	30
Q8	negative	30	0	0.97	1.00	0.98	30
	positive	1	29	1.00	0.97	0.98	30
Q9	negative	30	0	0.97	1.00	0.98	30
	positive	1	29	1.00	0.97	0.98	30
Q10	negative	28	2	1.00	0.93	0.97	30
	positive	0	30	0.94	1.00	0.97	30
Q11	negative	30	0	0.97	1.00	0.98	30
	positive	1	29	1.00	0.97	0.98	30
Q12	negative	29	1	1.00	0.97	0.98	30
	positive	0	30	0.97	1.00	0.98	30
Q13	negative	30	0	0.97	1.00	0.98	30
	positive	1	29	1.00	0.97	0.98	30
Q14	negative	30	0	1.00	1.00	1.00	30
	positive	0	30	1.00	1.00	1.00	30
Q15	negative	30	0	0.97	1.00	0.98	30
	positive	1	29	1.00	0.97	0.98	30
Q16	negative	30	0	0.97	1.00	0.98	30
	positive	1	29	1.00	0.97	0.98	30
Q17	negative	29	1	1.00	0.97	0.98	30
	positive	0	30	0.97	1.00	0.98	30
Q18	negative	30	0	0.97	1.00	0.98	30
	positive	1	29	1.00	0.97	0.98	30
Q19	negative	30	0	1.00	1.00	1.00	30
	positive	0	30	1.00	1.00	1.00	30
Q20	negative	30	0	1.00	1.00	1.00	30
	positive	0	30	1.00	1.00	1.00	30

7.1.2 Metrics from tests using a larger but unbalanced emission dataset

In this part of the experiment, 240 more negative samples are added to the validation dataset with the original 60 samples and the new validation dataset consists of 30 positive samples and 270 negative samples. The total size of the dataset has increased to 300 with 10% positive and 90% negative samples. Another test was carried out using this dataset for validation and evaluation.

As this is a highly unbalanced dataset even when the classifier merely guesses on the majority class, which is the class of negative samples, the accuracy score can still be as high as 90% but far less meaningful. As the ROC curves and associated AUC in this test also show negligible changes, we will mainly focus on the confusion matrix and related classification report, which is shown in Table 7.2.

In the table, the performances on precision and recall improved quickly during Q1 to Q6. In fact, the performances on the enlarged validation dataset are very similar to the balanced dataset in the last section, since around the 12^{th} query the classifications are almost correct for all the samples.

After further inspection, we found out that samples that are labelled as defects from the positive validation dataset are relatively difficult to classify as the pattern in the image is at the boundary between normal and defected. This is very similar to a negative training sample (but labeled as normal) in the dataset. As illustrated in Figure 7.3 the image on the left is the positive sample from the validation dataset and the one on the right is the negative sample from the training dataset which only shows a little less emission compared to the positive sample. This is an example that even a human annotator would have to think carefully before the decision of labelling the sample. From the aspect of testing and experiments, this situation can be good to increase the complexity of the dataset.

In order to corroborate our analysis, more tests were also done using the DAGM datasets, which contain more complex patterns for the classifier to understand during training. Results of this analysis are presented in the next subsection.

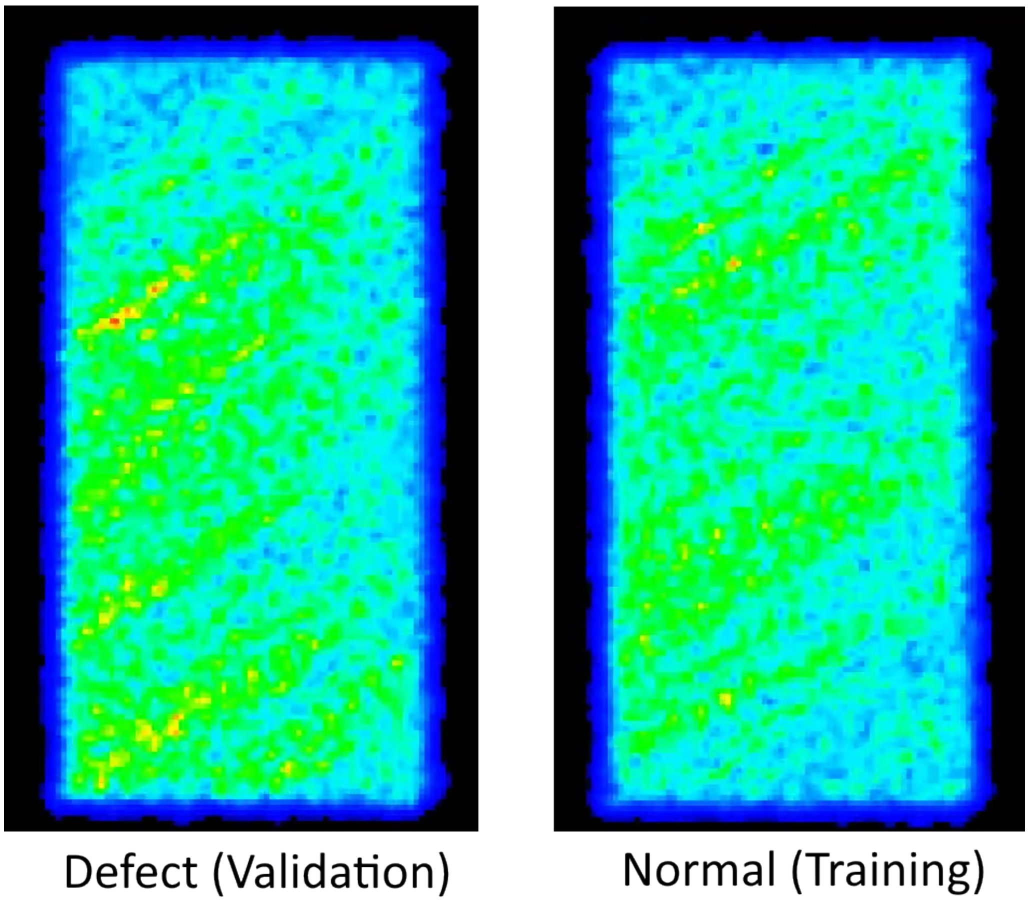

Figure 7.3: Comparing a positive sample from the validation (left) and a negative sample from the training dataset (right)

7.1.3 Validation and evaluation using the DAGM dataset

Considering the complexity of the patterns for the classifier, the DAGM datasets (refer to Figure 5.1 in Section 5.1) are used for further testing of the framework in order to investigate the capability of the classifier. This section illustrates the performance of the framework on the 6 patterns present in DAGM.

Table 7.2: Confusion matrices and classification figures for Q1 to Q20 using the enlarged unbalanced dataset.

		TN	FN	classification report			
Query	Label	FP	TP	precision	recall	f1-score	support
Q1	negative	262	8	1.00	0.97	0.98	270
	positive	0	30	0.79	1.00	0.88	30
Q2	negative	270	0	0.91	1.00	0.95	270
	positive	26	4	1.00	0.13	0.24	30
Q3	negative	270	0	1.00	1.00	1.00	270
	positive	1	29	1.00	0.97	0.98	30
Q4	negative	270	0	1.00	1.00	1.00	270
	positive	1	29	1.00	0.97	0.98	30
Q5	negative	269	1	1.00	1.00	1.00	270
	positive	1	29	0.97	0.97	0.97	30
Q6	negative	270	0	0.99	1.00	0.99	270
	positive	4	26	1.00	0.87	0.93	30
Q7	negative	270	0	1.00	1.00	1.00	270
	positive	1	29	1.00	0.97	0.98	30
Q8	negative	270	0	1.00	1.00	1.00	270
	positive	1	29	1.00	0.97	0.98	30
Q9	negative	270	0	1.00	1.00	1.00	270
	positive	1	29	1.00	0.97	0.98	30
Q10	negative	270	0	1.00	1.00	1.00	270
	positive	1	29	1.00	0.97	0.98	30
Q11	negative	270	0	1.00	1.00	1.00	270
	positive	1	29	1.00	0.97	0.98	30
Q12	negative	270	0	1.00	1.00	1.00	270
	positive	1	29	1.00	0.97	0.98	30
Q13	negative	269	1	1.00	1.00	1.00	270
	positive	0	30	0.97	1.00	0.98	30
Q14	negative	270	0	1.00	1.00	1.00	270
	positive	0	30	1.00	1.00	1.00	30
Q15	negative	270	0	1.00	1.00	1.00	270
	positive	0	30	1.00	1.00	1.00	30
Q16	negative	270	0	1.00	1.00	1.00	270
	positive	0	30	1.00	1.00	1.00	30
Q17	negative	270	0	1.00	1.00	1.00	270
	positive	0	30	1.00	1.00	1.00	30
Q18	negative	270	0	1.00	1.00	1.00	270
	positive	1	29	1.00	0.97	0.98	30
Q13	negative	269	1	1.00	1.00	1.00	270
	positive	0	30	0.97	1.00	0.98	30
Q20	negative	270	0	1.00	1.00	1.00	270
	positive	0	30	1.00	1.00	1.00	30

Pattern 1

As shown in Figure 7.4, Pattern 1 from the DAGM dataset starts with an AUC value of 0.619 and an accuracy around 60%. When reaching early convergence, the AUC value increased to 0.938 with an accuracy around 98% by query number 28. At the end of the final query 36, the AUC value is 0.986 and the accuracy is still around 98%.

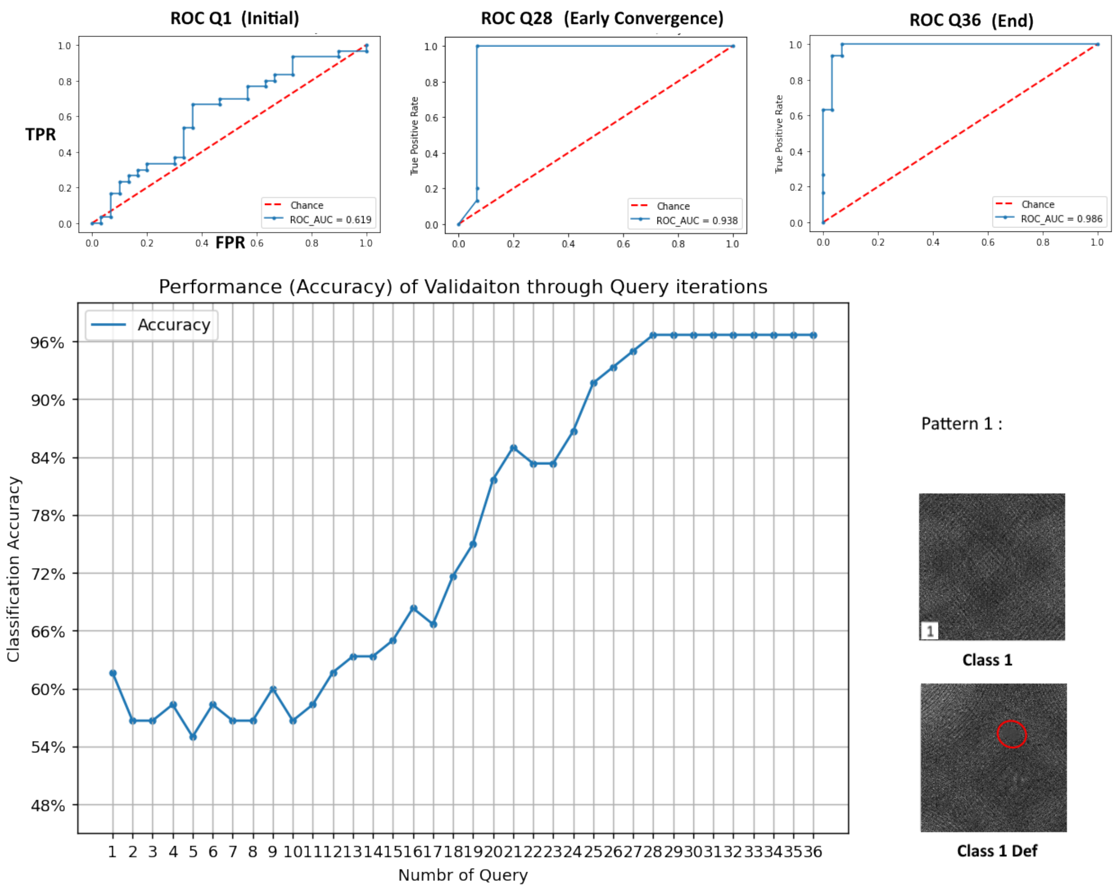

Figure 7.4: ROC curve and accuracy score of classification through the validation using DAGM pattern 1.

Pattern 2

As shown in Figure 7.4, Pattern 2 from the DAGM dataset starts with an AUC value of 0.636 and an accuracy around 60%. When reaching the early convergence, the AUC value increased to 1.000 with an accuracy around 98% by query number 16. At the end of the final query 36, the AUC value is 1.000 and the accuracy is still around 98%.

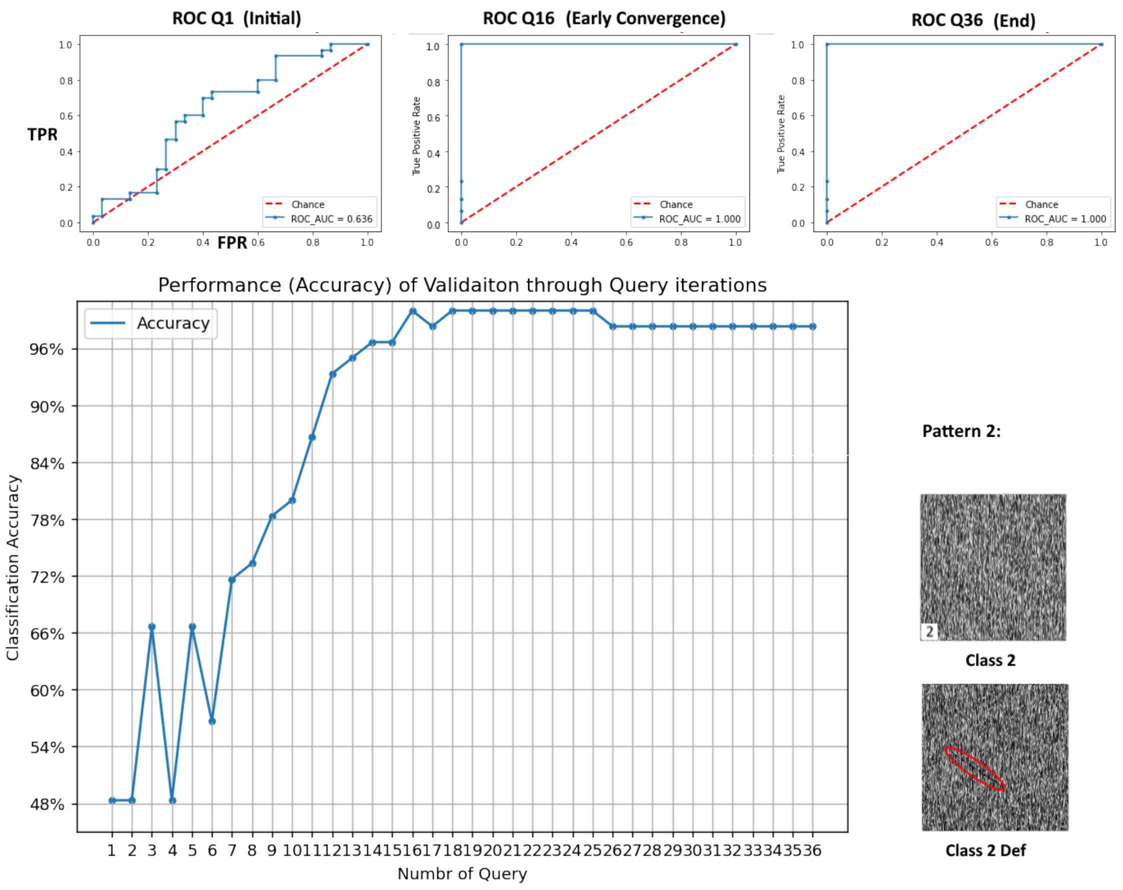

Figure 7.5: ROC curve and accuracy score of classification through the validation using DAGM pattern 2.

Pattern 3

As shown in Figure 7.6, Pattern 3 from the DAGM dataset starts with an AUC value of 0.576 and an accuracy around 60%. when reaching the early convergence, the AUC value increased to 0.996 with an accuracy around 96% by query number 24. At the end of the final query 36, the AUC value is 0.996 and the accuracy is over 96%.

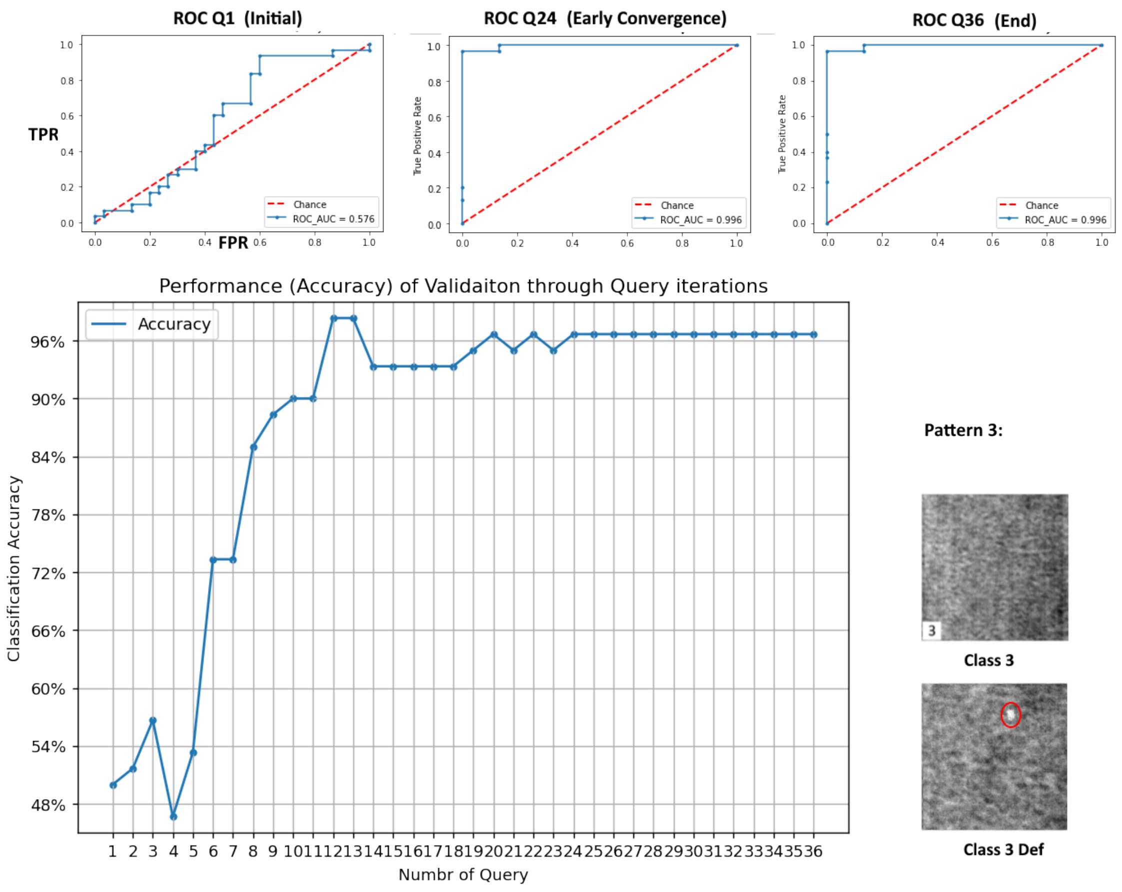

Figure 7.6: ROC curve and accuracy score of the classification through the validation using DAGM pattern 3.

Pattern 4

As shown in Figure 7.7, Pattern 4 from the DAGM dataset starts with an AUC value of 0.780 and an accuracy around 60%. when reaching the early convergence, the AUC value increased to 0.988 with an accuracy around 98% by query number 24. At the end of the final query 36, the AUC value is 0.991 and the accuracy is still around 98%.

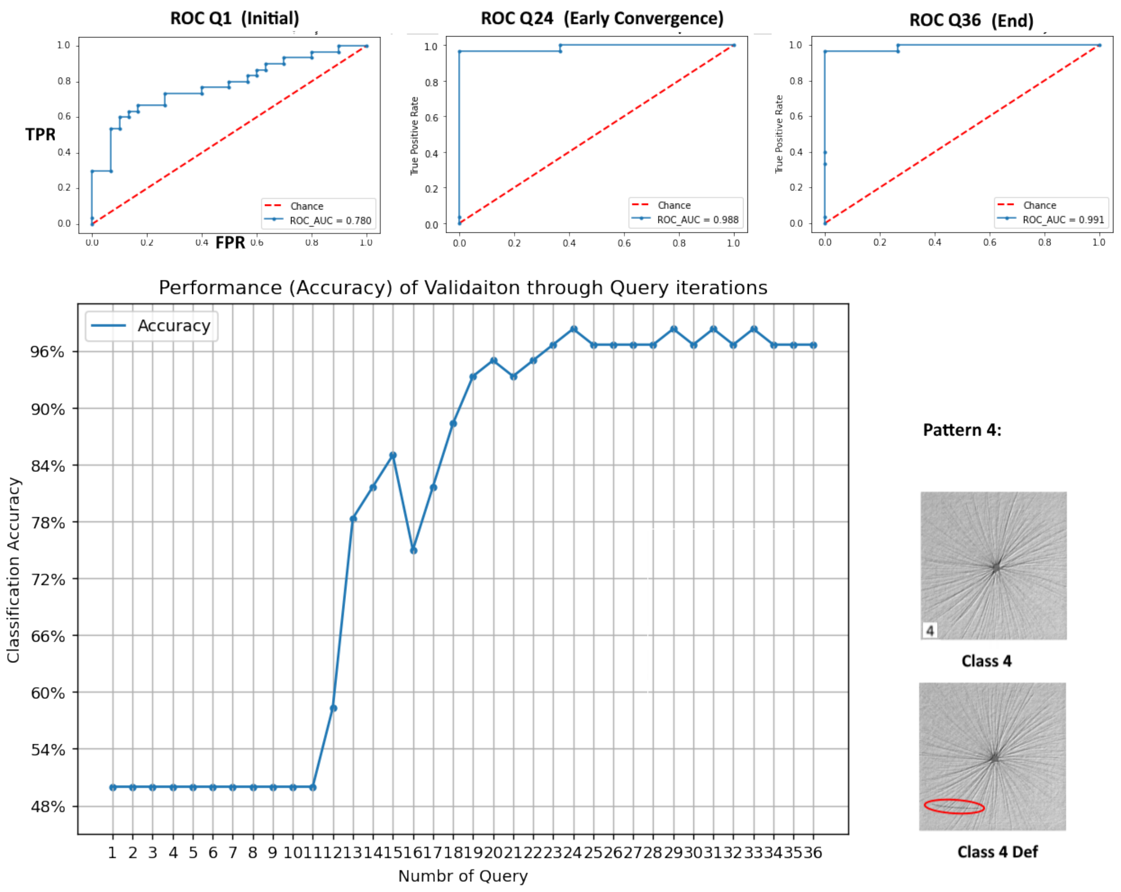

Figure 7.7: ROC curve and accuracy score of the classification through the validation using DAGM pattern 4.

Pattern 5

As shown in Figure 7.8, Pattern 5 from the DAGM dataset starts with an AUC value of 0.531 and an accuracy around 60%. when reaching the early convergence, the AUC value increased to 0.999 with an accuracy around 97% by query number 12. At the end of the final query 36, the AUC value is 1.000 and the accuracy is still around 97%.

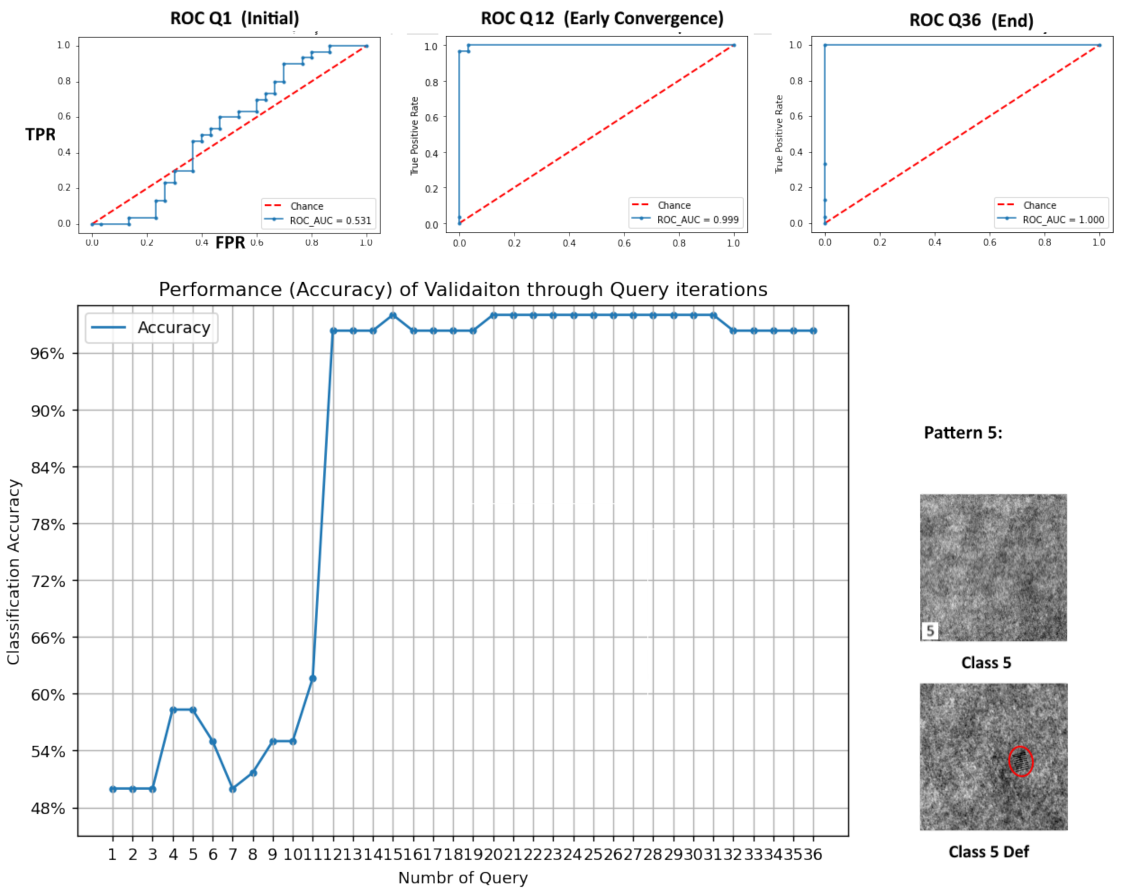

Figure 7.8: ROC curve and accuracy score of the classification through the validation using DAGM pattern 5.

Pattern 6

As shown in Figure 7.9, Pattern 6 from the DAGM dataset start with an AUC value of 0.914 and an accuracy around 60%. when reaching the early convergence, the AUC value increased to 1.000 with an accuracy around 98% by query number 12. At the end of the final query 36, the AUC value is 1.000 and the accuracy is still around 98%.

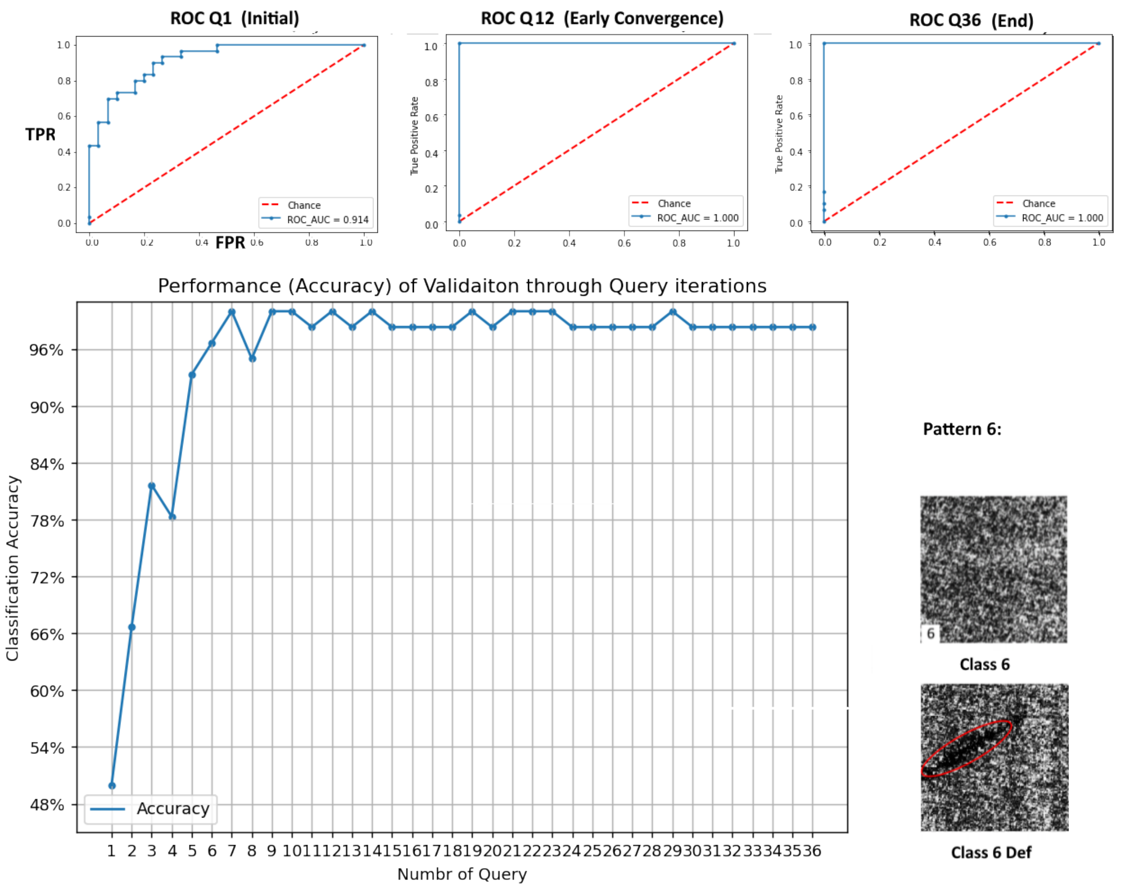

Figure 7.9: ROC curve and accuracy score of the classification through the validation using DAGM pattern 6.

Summary of tests on DAGM at early convergence

Table 7.3 shows the detailed classification report at the early convergence queries for each pattern in the DAGM dataset with training length (epochs) for the query iteration. From the results, Pattern 1 is the most difficult pattern to be learned by the model as it requires the most number of epochs in each query iteration and the most number of queries to reach the convergence at a relatively good level of performance. Pattern 6 is the easiest pattern out of the total 6 DAGM patterns as it only requires 12 queries to reach high performance in early convergence. In fact, from the examples of pattern 1 samples, we find that pattern 1 in Figure 7.4 is difficult to be identified even by the human eye as the object in the image is small and blurred with a very noisy background.

On the other hand, pattern 6 with similar appearance to the emission images (but in grayscale) can be learned by the classifier relatively easily. Overall, regardless of the complexity of the patterns, all the 6 patterns from the DAGM dataset can be classified by our approach with an accuracy over 96% which proves the capability of the framework is relatively good and the early convergence also proved that our approach can potentially save labelling time and computational resources.

Table 7.3: Detailed classification report at the early convergence query for each pattern in DAGM with training length (epochs) for the query iteration.

		TN	FN	Classification Report					
Pattern	Label	FP	TP	Precision	Recall	F1-score	Support	At Query	Epochs per Query
1	negative	28	2	1.00	0.93	0.97	30	Q28	400
	positive	0	30	0.94	1.00	0.97	30		
2	negative	29	1	1.00	0.97	0.98	30	Q16	150
	positive	0	30	0.97	1.00	0.98	30		
3	negative	29	1	0.97	0.97	0.97	30	Q24	350
	positive	1	29	0.97	0.97	0.97	30		
4	negative	30	0	0.97	1.00	0.98	30	Q24	100
	positive	1	29	1.00	0.97	0.98	30		
5	negative	30	0	0.97	1.00	0.98	30	Q12	200
	positive	1	29	1.00	0.97	0.98	30		
6	negative	30	0	1.00	1.00	1.00	30	Q12	75
	positive	0	30	1.00	1.00	1.00	30		

7.2 Discussion on Stopping Criteria

So far, the performance of this framework has been evaluated using patterns from emission images and further tested using the 6 patterns from the DAGM dataset to validate the capability of the model in detecting more complex patterns. We have seen improvements in our performance metrics during the validation of the framework over subsequent interactions of training and active learning. Considering the higher level of overall accuracy the framework can achieve, we consider this as an indication of the potential for use in automatic data labelling.

To justify the training length which is the number of epochs for the active learning of the classifier in each iteration, there are two conditions that should be meet in order to ensure the proper training of the classifier and avoid negative effects which would potentially be detrimental for the performances in the next iteration and ultimately in the overall training process, which are listed in what follows.

1. The training length should be sufficient to reach the relative convergence of the performance in the current iteration. This can be justified according to several performance metrics. In practice, the first prioritised information for this is the training loss as it directly assesses how the classification model fits the current training data. The number of training epochs should be greater that the epoch where the early convergence occurs to optimise the current training data samples. An insufficient training length would result in more queries required to reach a certain performance. In that situation, the human annotator has to label the samples in the additional queries. And even so the classifier may still not be able to reach convergence when the sampling budget is exhausted.
2. The second condition for the training length is to avoid overfitting. Depending on the complexity of the model, overfitting may not always happen but it is still an important factor that should not be ignored. A simple but effective solution to prevent overfitting is the early stopping approach, which uses both the training loss and the validation loss to monitor the training process. Keeping this in mind, the number of epochs in each query iteration should be limited to not allow overfitting to happen.

It turns out that, depending on the complexity of the samples, the suitable training length can differ: patterns with higher complexity will usually require a larger number of epochs in the training process before the classifier achieves an relatively good performance which verifies the above two conditions. As further investigation on the stopping criteria is more relevant to future work, additional discussion will be presented in the future work section of this chapter, which is section 7.4

7.3 Labeling Using Semi-Supervised Learning with Human-in-the-Loop

Up to this point, we have presented an approach that employs a CNN based classifier in combination with transfer learning and active learning strategies. The developed classifier has served as a robust foundation, setting the stage for further advancements in the labeling mechanism which involves leveraging semi-supervised learning techniques with the integration of human-in-the-loop features. This approach aims to augment and refine the labeling process, capitalising on the strengths of both automated learning and human supervision to further enhance the accuracy of the labelling work, the performance of the model and the applicability of the proposed approach in the domain of additive manufacturing defect detection. In addition, this section focuses specifically on the auto-labelling block illustrated in Figure 4.3. In this block, the trained classifier generates labels for the samples through automatic prediction, and these labels are then selectively supervised with the human-in-the-loop (HITL) feature.

7.3.1 Proposed Approach to Address the Class Imbalance Issue

In the original work of [25], Random Undersampling (RUS) and Random Oversampling (ROS) were used to create the balanced dataset for training, validation and testing purposes. These methods, while a commonly used technique, are not the most up-to-date or advanced method for addressing class imbalance in machine learning.

While SMOTE is a more up-to-date approach, it may not be the best choice for oversampling image data, especially in the context of additive manufacturing. Originally designed for oversampling tabular data, SMOTE may not fully capture the complex patterns and structures present in image data, particularly when the images are high-dimensional and contain intricate patterns, textures, and structures. This limitation is especially pertinent in images from additive manufacturing processes, where such complexity is common. Basic linear interpolation between data points may not fully capture the complex relationships in image data. In particular situations, image pixels have spatial correlations, and the local neighbourhood of a pixel is critical for interpreting its value. As SMOTE does not consider these spatial relationships, it has a risk of generating unrealistic synthetic images and this can be problematic if the quality of synthetic images is crucial, as it is here. Furthermore, the implementation of our CNN based model for image classification tasks in this case study has built-in mechanisms for conducting data augmentation during the data acquisition and model training process. Leveraging these built-in mechanisms can be a more effective way to oversample the image data of the minority class to handle the data imbalance problem.

For the above reasons, we will not consider synthetic data created by SMOTE. In addition to SMOTE, there are many generative algorithms available, such as Generative Adversarial Networks and Stable Diffusion. However, these approaches require a lot of images to train on. In this use case this cannot be done with any reliability as the number of images are very small and the problem will point back to the “small data challenge” that we are already addressing in our research.

For an alternative method of oversampling, we present our approach that combines uncertainty sampling with image data augmentation to effectively address class imbalance. This method places a strong emphasis on selecting the most informative samples, by identifying instances where the model exhibits uncertainty in its predictions. These informative samples are then systematically re-sampled using image data augmentation techniques, including transformations such as rotation, scaling, flipping, and cropping according to the relevant data structures. The objective is to generate a diverse set of new samples while preserving spatial correlations and image quality. This approach stands out as more preferable compared to synthetic image data generation, especially in the context of additive manufacturing, where data reliability and fidelity are extremely important.

1. Given an actively selected batch of samples L_{batch} with a number of samples n_{batch} by the trained classifier C_i using uncertainty sampling, so L_{batch} represents a batch of the most informative samples from the pool of samples N_{pool}. It is worth noting that N_{pool} can also represent any collection of samples that need to be processed, such as a collection of pseudo-labels generated by C_i. However, for clarity, it is simply described as a pool of samples.

2. The size of n_{batch} is decided by the sampling budget of the human annotator, for example, 10% of the total samples from N_{pool} or any specific number decided by the annotator.

3. Assuming labeling by the human annotator is correct and treated as ground truth, the samples from n_{batch} samples are labeled by the human annotator. This results in two collections: L_{minor} for labeled samples of the minority class, with a sample size of n_{minor}, and L_{major} for labeled samples of the majority class, with a sample size of n_{major}. For additional clarity, L_{minor} represents the collection of samples belonging to the minority class, and n_{minor} denotes the actual number of samples in this collection. Similar explanations can be applied to L_{major} and n_{major}.

4. In the case of balancing the number of samples between the minority and majority classes, we prioritise oversampling the minority samples because there are usually limited or insufficient examples of minority cases, such as defects in additive manufacturing.

5. To prevent excessive duplication of samples and preserve dataset integrity, oversampling will only occur once per sample. Therefore, the number of mi-

nority samples after oversampling n_{over} will be under the condition that $n_{over} \leq n_{minor}$

6. As illustrated earlier in this section, image augmentation techniques such as flipping, rotating, and resizing will be employed for oversampling the selected samples.

7. If $2 \times n_{minor} > n_{major}$ which means not all the minority samples need to be oversampled to achieve the balance. in this case, only $n_{over} = n_{major}$-n_{minor} samples need to be oversampled and the collection of samples that need to be oversampled are selected by classifier C_i using uncertainty sampling from the minority samples.If use L_{over} to stand for the collocation of the resulting samples by oversampling, the resulting balanced collection of the minority is $L_{balminor} = L_{minor} + L_{over}$ and for the balanced majority, $L_{balmajor} = L_{major}$

8. If $2 \times n_{minor} = n_{major}$ which means the batch will be balanced after applying oversampling to all the samples from the minority. If use L_{over} to stand for the collocation of the resulting samples by oversampling, the resulting balanced collection of the minority is $L_{balminor} = L_{minor} + L_{over}$ and for the balanced majority, $L_{balmajor} = L_{major}$

9. If $2 \times n_{minor} < n_{major}$ which means the number of minority samples will still be less than the number of majority samples even after applying oversampling to all the samples from the minority. In this case, after oversampling all the minority, $L_{balminor}$ is still generated similar to the previous conditions that collection of the minority after re-sampling is still $L_{balminor} = L_{minor} + L_{over}$ where L_{over} is generated by oversampling all the samples in L_{minor}. After this, undersampling will be applied on L_{major} to balance the number of samples between $L_{balmajor}$ and $L_{balminor}$. During the process, uncertainty sampling strategy is applied to L_{major} to select the requied n_{under} samples from L_{major}, where $n_{under} = 2 \times n_{minor}$. If use L_{under} to stand for the collocation of the resulting samples by undersampling the majority sampls, the resulting balanced

collection of the majority is $L_{balmajor} = L_{under}$.

10. After the oversampling or undersampling process, the balanced training batch $L_{balmajor} + L_{balminor}$ are added to the original training dataset T_i. Then, the updated training dataset can be used for the updating the deep learning model of the classifier C_i or training new models.

The overall process of re-sampling can be summarised as Algorithm 5. This algorithm will also be used in the labeling method that will be intrduced in the next section.

Algorithm 5 re-sampling with active selection and augmentation

```
1: procedure RE-SAMPLING TO BALANCE THE LABEL OF ANNOTATED IMAGE SAMPLES()
2:     Given a batch of samples $L_{batch}$ = Trained Classifier $C_i$ actively selected from a pool of samples $N_{pool}$
3:     Assuming labeling by the human annotator is correct and treated as ground truth, after labeling by annotator $L_{batch}$ contains 2 collections: $L_{minor} + L_{major} = L_{batch}$
4:     re-sampling $L_{minor}$ and $L_{major}$ to create a balanced data group with conditions:
5:     if $2 \times n_{minor} < n_{major}$ then
6:         $n_{over} \leftarrow n_{major} - n_{minor}$
7:         $L_{over} \leftarrow C_i$ actively selected $n_{over}$ samples from $L_{minor}$
8:         $L_{balminor} \leftarrow L_{minor} + \text{Augmented}(L_{over})$
9:         $L_{balmajor} = L_{major}$                ▷ No option required on $L_{major}$
10:    else if $2 \times n_{minor} = n_{major}$ then
11:        $L_{over} \leftarrow L_{minor}$                ▷ all $L_{minor}$ will be oversampled
12:        $L_{balminor} \leftarrow L_{minor} + \text{Augmented}(L_{over})$
13:        $L_{balmajor} = L_{major}$                ▷ still no option required on $L_{major}$
14:    else if $2 \times n_{minor} < n_{major}$ then
15:        $L_{over} \leftarrow L_{minor}$                ▷ all $L_{minor}$ will be oversampled
16:        $L_{balminor} \leftarrow L_{minor} + \text{Augmented}(L_{over})$
17:        $n_{under} = 2 \times n_{minor}$
18:        $L_{under} \leftarrow C_i$ actively selected $n_{under}$ samples from $L_{major}$    ▷ undersampling on $L_{major}$
19:        $L_{balmajor} = L_{under}$
20:    end if
21:    $L_{update} \leftarrow L_{balmajor} + L_{balminor}$    ▷ $L_{update}$ is the collocation of samples after re-sampling that can be used to update the original training dataset
22: end procedure
```

7.3.2 Pseudo-labelling with the Human-in-the-Loop Feature

Early in Section 2.2.6, Pseudo-labelling have been mentioned in the work of [70]. However, It is important to note the difference between the technique explored in our study and the one proposed by [70] in which labeled and unlabeled data are used simultaneously during the training schedule and the pseudo-labels are recalculated after every weight update. Alternatively, in our study adopts the concept that use the trained classifier to generate pseudo-labels but rather than directly update the model using the pseudo-labels, we involved human supervision to supervise a selection from the pseudo-labels according to the desired labelling budget specified by the annotators. During this process, the re-sampling algorithm 5 will also be applied to maintain the balance of the annotated samples that then will be used to enlarge the training dataset.

As previously highlighted in Section 2.2.6, three drawbacks associated with pseudo-labelling have been identified, namely, a reliance on the initial model quality, challenges related to class imbalance, and limited guidance. In our study, we have already addressed the first drawback of pseudo-labelling where we trained the model using Transfer Learning, fine-tuning, and active learning based on the uncertainty sampling strategy. This approach has provided us with an initial classifier model of reliable quality, effectively addressing the first drawback of pseudo-labelling listed in Section 2.2.6. The next stage of our approach involves incorporating pseudo labels into the model's further fine-tuning. This stage is also where we develop re-sampling strategies (refer to section 7.3.1) to solve the class imbalance problem while introducing human-in-the-loop features into the process to update the model and address the issue of limited guidance.

In this subsection, we will illustrate the human-in-the-Loop (HITL) features, which will be incorporated into the Pseudo-Labelling process to address the third drawback outlined earlier. Specifically, The purpose of this HITL feature can be described in two key points:

1. we start the training by active selection (uncertainty sampling) for the initial

training and perform multiple rounds of such kind of training iteration until the classification accuracy converges at a desired level for example, around 97%.

2. The HITL feature provides responses to active queries concerning uncertain samples, enhancing the reliability of labels assigned during the Pseudo-Labelling process;

3. It functions as a corrective mechanism for potential labeling errors, reducing the risk of noise amplification issues within the dataset.

By integrating human expertise and oversight into the labelling process, HITL ensures the accuracy and quality of labeled data, which is important for a machine learning based applications in AM for the defect classification and labelling task and it can further refine the performance and robustness of the model. the process can be summarised as algorithm 6. The procedure in Algorithm 6 can be repeated as iterations if more human supervision is required to improve the performance.

Algorithm 6 Process of Pseudo-labelling with HITL

```
1: procedure PSEUDO-LABELLING WITH HITL
2:     Given a trained classifier $C_i$ using $T_i$    ▷ $T_i$ is the orignal traning dataset
3:     $L_{pseudo} \leftarrow C_i$ to prediction on $N_{pool}$
4:     $L_{batch} \leftarrow C_i$ actively selected from $L_{pseudo}$    ▷ using uncertainty sampling
5:     $L_{HITL} \leftarrow$ Human annotator assign label to $L_{batch}$    ▷ Assuming annotation
   by the human annotator is correct and treated as ground truth
6:     $L_{update} \leftarrow$ apply algorithm 5 on $L_{HITL}$    ▷ re-sampling using algorithm 5 to
   create a balanced batch of data
7:     $T_i \leftarrow T_i + L_{update}$    ▷ update $T_i$ by adding $L_{update}$ to $T_i$
8:     The updated $T_i$ then can be used to update $C_i$ or used to train new models.
9: end procedure
```

7.4 Next Steps

There can be a trade-off between classification accuracy and human workload during the active query based training stage. This means we can consider an early stop in the training to save the budget of samples to be labeled by annotator at the early convergence stage of the training process with a cost of slightly lower accuracy in the auto-labeling stage. However, this may also potentially result in a higher risk of a self reinforcement that determines deterioration of the quality of labelling results if we want to use the samples for the auto-labeling to further apply fine-tuning on the classifier. Continuing from the discussion in section 7.2, the stopping criteria is used to decide the point where the active training process stops and the automatic labeling can begin. As the training process of the classifier is query-based, the stopping point for active learning and training should happen after a certain number of query iterations.

In the current setup, we decided to set the stopping criteria according to the performance metrics from the validation such as F1-score, recall, precision and accuracy score. The selection of metrics should be flexible according to the structure of the available data. The stopping criteria should be reached when the performance metrics no longer improve significantly above a certain threshold which means the training process is getting close to optimising the value of training in the samples, and even further training using more samples from the pool of data will not significantly update the classifier.

As there can be more data generated from new manufacturing processes, the pool of samples can be updated. Thus, there can be samples to be labeled carrying new features that require further training and update of the model for sufficiently accurate classification. This is when the training needs to be reactivated. In this case, the validation dataset should also be updated with the new samples to match the updated classifier. As the labeling process during training is based on active queries, augmenting the validation dataset should also be done in an active learning style. Similarly to the learning and training process, in order to add new samples in the validation dataset, we can let the classifier apply active sample selection using the same query strategy that have been used during training, but this time the selected samples will only be annotated for the correct label and moved to the validation dataset to be further used for the evaluation of the classifier and justify the new stopping criteria of the active query and training for the updated classifier. We leave this deeper analysis and experimentation on the impact of different stopping criteria as a direction of investigation for future work. In addition, we considered the challenges and opportunities arising when involving more data samples from manufacturing in the aspect of updating the validation set by further leveraging the active learning strategy. This can also be considered in future work and development.

7.5 Chapter Summary

In this chapter we have validated and evaluated the performance of the framework using the emission images dataset from AM as well as the patterns from the DAGM dataset to inspect the performance of the labeling work done by the classification model when processing the datasets with different structures and various level of complexity. During this process, relevant performance metrics have been computed and analysed. We have also commented on on the metric used and the ability to generalise to different patterns and additive manufacturing tasks.

Up to this point, we have finalised the proposed adaptive human-in-the-loop framework by utilising the advantages of the combination of deep learning with Active Learning to produce relatively good quality labelled data within a limited labelling budget for training and fast convergence. We have also discussed the stopping criteria for each query iteration and the active training process to decide the starting point for efficient automatic labeling. In the next chapter we will conduct an additional case study on defect detection using a different dataset, identified earlier from [25].

Chapter 8

A Case Study on Defect Detection in an Additive Manufacturing Dataset

8.1 Introduction

This case study is based on the findings of the systematic review conducted on open datasets within the domain of additive manufacturing for machine learning applications which is described in Chapter 3. Subsequently, this chapter will provide a detailed illustration of the deep learning model developed in the thesis for classification, along with the relevant methodology of training and labeling as well as the execution of associated experiments. All of this is done utilising the dataset sourced from the scholarly work documented in the article by Westphal *et al.* [25] within the Additive Manufacturing journal published by Elsevier, the only realistic AM dataset usable for such experiments, as described in the conclusions in Chapter 3. That dataset was developed from a collection of discrete images captured by a camera above the powder bed area during the additive manufacturing process. More specifically, the image dataset focuses on the print area of the manufactured parts, which were originally extracted from a recorded video dataset.

In the case study in this chapter, we will perform an investigation into the performance characteristics of a pre-trained VGG16 convolutional neural network model when applied to an evaluation on a dataset associated with the area of additive manufacturing. This study stands as a valuable contribution to the field of additive manufacturing by proposing the incorporation of active learning methodologies with the intent of enhancing the model's efficacy in handling the specific dataset nuances, which are particularly pronounced within the manufacturing domain, owing to the presence of inherent noise and data sparsity.

The experimental methodology adopted in this case study demonstrates an approach that requires a substantially reduced volume of training data in comparison to the overall dataset size. Moreover, this approach provides a tangible illustration of the impact of active learning techniques. Consequently, this study introduces the concept of a HITL mechanism that effectively reduces the requisite dataset dimensions necessary for classifier training while concurrently facilitating the generation of additional labeled data instances.

To further clarify the method used in this case study, the illustration of steps and the corresponding approaches presented in the previous chapters are listed below:

1. The first step, where the model implementation (section 8.3.1) and tuning (section 8.3.2) are illustrated. The experiment began with a deep learning model identical in architecture to that illustrated in Chapter 5. The detailed architecture of the model is referenced in Figure 5.3. Transfer learning using ImageNet data was applied to the model. The only difference is that the input dataset this time is from the dataset specific to this case study.

2. The second step involved applying the training approach outlined in Chapter 6, specifically the training approach with active sample selection. This step aimed to investigate the training performance on the dataset specific to this case study. Further details on the application of the approach in this study can be found in Section 8.3.4 and Section 8.3.5.

3. The third step involves conducting experiments that incorporate semi-supervised learning with human-in-the-loop features for labeling and enlarging the training dataset. During this process, resampling methods are used to address class imbalance issues and balance the newly generated labels before adding them to the training dataset. The approach used is outlined in section 7.3 of Chapter 7.

8.2 Preliminary work

Selective Laser Sintering (SLS) [131] is a subtype of PBF techniques in Additive Manufacturing in which a substrate comprised of powder, including materials such as polymer, resin, or metal, undergoes controlled transformation through the application of a high-power directional laser. This process will conduct partial sintering or even complete melting, leading to the formation of a consolidated layer from the powdered material. As introduced earlier in the thesis in Section 2.1.4, in 2021 Westphal *et al.* [25] presented work that employs convolutional neural networks as a machine learning approach for defect detection and visualisation within the context of Selective Laser Sintering.

Within that work, the authors reported that their best results in defect classification were obtained through a CNN model based on VGG16 architecture. Importantly, the dataset used for their classification experiments is publicly accessible. It is worth noting that this dataset originates from a polymer-based additive manufacturing machine not a metal powder bed system. However, this is still a good choice for experimentation from a data analytics perspective, as process prediction and control are required in both the polymer and metal additive manufacturing domains.

8.2.1 The dataset

The open access dataset published in the work by Westphal and Seitz is originally from an initial compilation of 9,426 raw images captured within the powder bed environment. These images involve a variety of content, which are irregularities, sintered elements, sintered elements with irregularities, and blanks. Some representative examples of these image categories are shown in Figure 8.1. In a further process, extraneous images, which are those affected by recording anomalies and unfavorable light reflections are eliminated in order to refine the dataset and resulted in 8,514 initially processed images. Moreover, in the original paper, the authors illustrated another phase of further image processing, aiming to generate 4,000 images with simplified labels from the pool of the 8,514 initially processed images. This process was conducted by utilising only the pertinent information contained within the images and supplying the CNN models with pixels of interest. The image processing procedure can be summarised into three distinct steps:

1. Firstly, black stripes present along the image edges were removed due to the fact that these areas contain no pertinent information for the intended classification task. This step was conducted with the purpose of avoiding unnecessary computational overhead and to reduce the consumption of computational resources to optimise the overall efficiency of the training and classification tasks.

2. Secondly, a resizing operation was performed to ensure that the smaller edges of each image measured precisely as 180 pixels in length. This standardisation of image size was aimed to create uniformity across the samples and to maintain the consistency in the dimensions of the images for further analysis and processing tasks.

3. Thirdly, another operation was conducted to extract a centered square area of 180 by 180 pixels in the images, as shown in Figure 8.2. This step was aimed to isolate a focused region of interest within each image, enhancing the

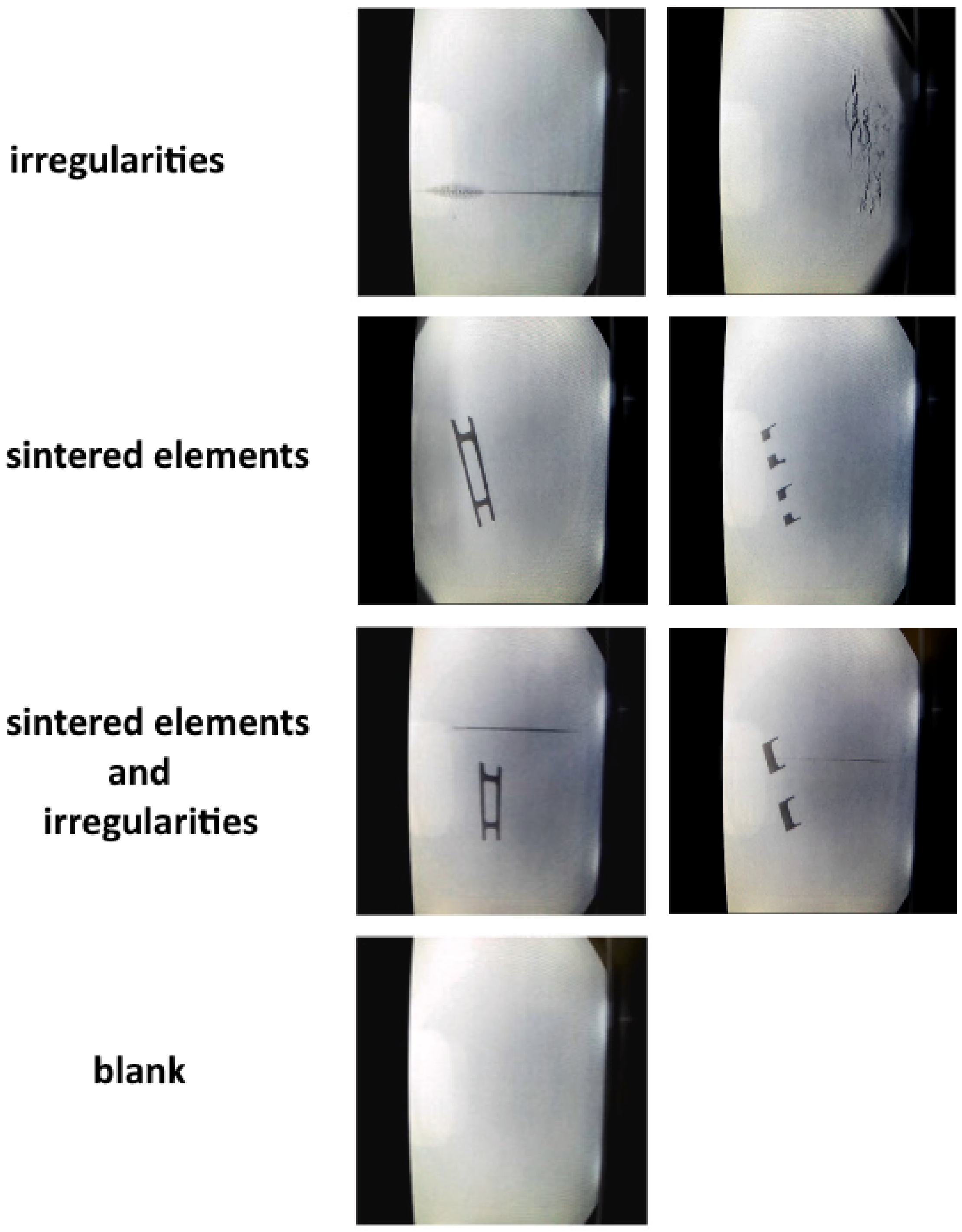

Figure 8.1: Sample images for 1.irregularities, 2.sintered elements, 3.both sintered elements and irregularities, 4.blanks.

relevance of the extracted data for subsequent tasks.

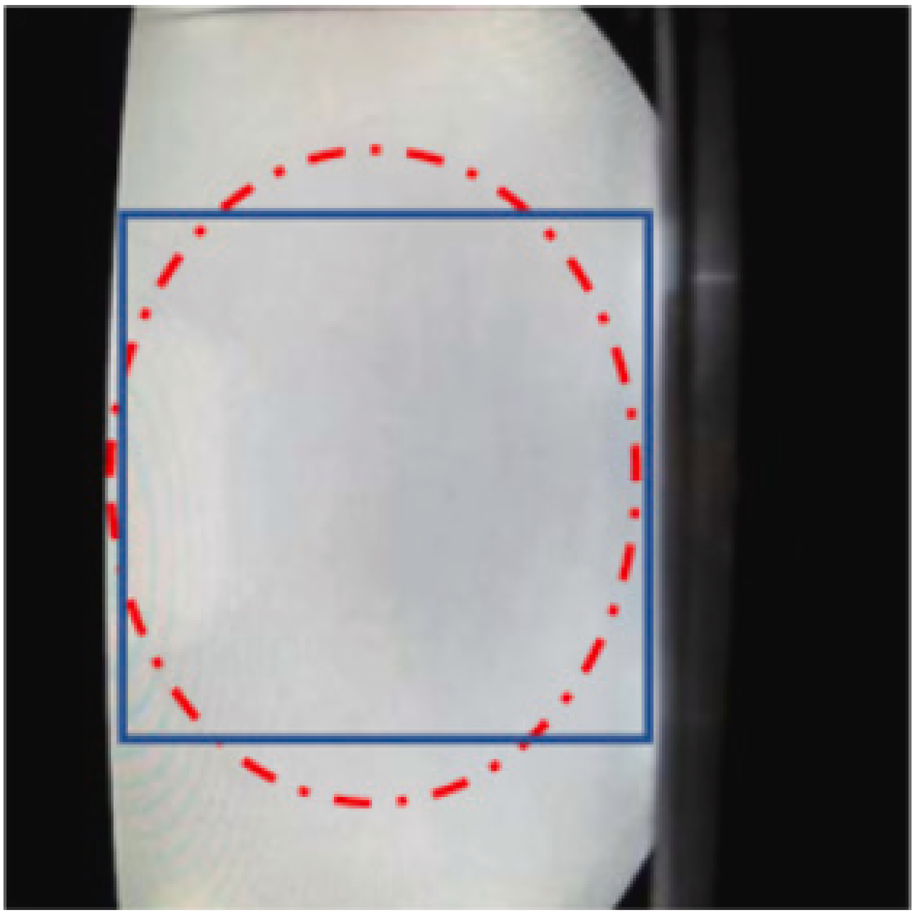
Figure 8.2: Sample of processed area within an image

The dataset of 4,000 processed images was then manually categorized, dividing them into two distinct classes for defect classification related tasks. The dataset exhibits a well-defined separation into three balanced subsets for training (2,000 images), testing (1,000 images) and validation (1,000 images) purposes. Figure 8.2 shows examples of the processed images involved in this dataset.

Having this openly available image dataset as a foundation, we implemented our approach that also utilises a VGG16-based CNN deep learning model that had been developed using our own design that closely aligns the architectural framework detailed in Chapter 5 of this Thesis. Our primary research objective was to investigate the possibilities offered by transfer learning, active learning, and their synergy, further augmented by the combination of human-in-the-loop features and semi-supervised learning methodologies. These explorations were conducted within the domain of defect classification and sample labeling that are specific to the applications in Additive Manufacturing.

8.3 Method

In the original research paper by Westphal *et al.* [25], a series of three experiments were conducted to demonstrate the efficacy of transfer learning and fine-tuning methodologies within the context of the Additive Manufacturing (AM) domain. As discussed in Section 2.1.4, the highest classification performance was achieved using a VGG16-based CNN model, which served as the baseline for our own subsequent investigations. Consequently, we did not adopt any of the more recent models such as ResNet for our study.

It is important to note that during our examination of the original paper, we identified an error in the results presented for the second experiment, as detailed in Table 2.2. More specifically, the numbers within the confusion matrix obtained from their VGG16-based model, which also constituted their best-performing results, did not align with the number of samples in the validation dataset. To address this issue, we communicated with the original authors of the paper in order to bring the error to their attention. Later in their response, they provided us with a revised set of results, which are presented in Table 8.1, along with relevant instructions detailed in the caption of the table.

8.3.1 Implementing the CNN model on the Dataset

To conduct our experiments, we also employed a VGG16-based classifier based on our previous work that has been illustrated in Chapter 5 which proved to be accurate in the task of defect classification on images generated from emission monitoring during the additive manufacturing process. All of the 6 DAGM datasets with values are over 95% in the accuracy of all the classification task. The implementation of our classification model relies on transfer learning in which 13 convolutional layers from a pre-trained VGG16 model are used for feature extraction and the weights in these layers had been trained using ImageNet data. After the convolutional layers, 2 dense layers with ReLU activation function are added followed by 1 dense layer as the output layer using Sigmoid as the activation function since the targeted dataset is divided into 2 classes for binary classification.

After the constriction of the architecture of the model, fine-tuning is applied using the targeted dataset which as mentioned earlier, was created from SLS powder bed images to train the model and significantly improve the performance of the model in the classification task. Based on this classifier, we apply active learning during the training process to further reduce the number of samples required for the training of the model during the tuning process, similar to the active learning approach that illustrated in Chapter 6 which is based on the uncertainty sampling method.

8.3.2 Tuning of hyperpapmeters

The process of hyperparameter tuning involves the adjustment of key parameters that significantly impact the performance of a classification model. In our experimentation, we explored various combinations of optimisers, learning rates, batch sizes, and training epochs to optimise our model's performance. Specifically:

1. Optimisers: We assessed the effectiveness of three different optimisers which are Adam, SGD, and RMSprop when used in conjunction with learning rates in a range from 10^{-2} to 10^{-5};

2. Batch Sizes: We conducted training with different batch sizes, specifically 4, 32, and 64, to assess their impact on model performance;

3. Training Epochs: We varied the number of training epochs across three values of 30, 60, and 120 to understand the trade-offs in training duration vs. model accuracy;

4. Cost Function: Binary cross-entropy served as the cost function in all of our experiments.

Moreover, to combat overfitting, we introduced weight regularisers to the two dense layers employing the ReLU activation function, as previously mentioned. We applied weight decay regularisation, also referred to as L2 regularisation, which calculates the sum of squared weights. The hyperparameter tuning for weight decay regularisation spanned a range from 10^{-1} to 10^{-4} and was tested multiple times until overfitting issues no longer surfaced during training and validation. This tuning process aimed to keep a balance between model complexity and generalisation ability, ensuring the model's robustness.

8.3.3 Comparison with the orignal paper

Before our investigation into applying active learning, we performed a preliminary evaluation of our classification model's performance on this data. This evaluation focused solely on the aspects of transfer learning and fine-tuning. In this initial testing phase, we employed the entire training dataset to fine-tune the classification model. Subsequently, we compared the output with the baseline performance. This preliminary assessment allowed us to establish a baseline reference point for the capabilities of our classification model, predicated solely on the use of transfer learning and fine-tuning techniques. It served as a valuable starting point for our subsequent investigations into the integration of active learning and other advanced methodologies to further enhance the performance of the model.

Following an extensive tuning process encompassing various hyperparameter combinations mentioned earlier, we identified the best-performing configurations for the three different optimisers, which are presented in Table 8.1. This table also includes the classification results on the validation dataset, providing a basis for comparison with the baseline results obtained from the original work by [25].

The initial tests served as a crucial assessment of the adaptability of our approach to the specific dataset in question. The results demonstrate that all three optimisers can achieve validation accuracy levels approaching 98%, indicating that our classification model is exceptionally well-suited to this dataset.

Table 8.1: Best performing hyperparameters for each optimiser, performance results on the validation set. Results marked '*' are updates provided directly to us by the authors of [25] in response to us pointing out errors in their original paper. An author correction to the copy of record is now underway.

Experiment: Optimiser, learning rate	Batch	Epochs	Confusion matrix		Accuracy	Precision	Recall	F1-Score	AUC
Baseline	64	30	496	4*	0.977*	0.992*	0.963*	0.977*	0.993
			19	481					
SGD, lr=0.01	4	60	483	17	0.979	0.967	0.992	0.979	0.998
			4	496					
Adam, lr =0.00001	4	120	490	10	0.988	0.980	0.996	0.988	0.998
			2	498					
RMSprop lr =0.00001	4	60	485	15	0.982	0.971	0.994	0.982	0.997
			3	497					

Furthermore, the results indicate that a smaller batch size, such as 4, yields superior performance. This observation can be attributed to smaller batch sizes necessitating more frequent weight updates during training. Consequently, the model can adapt its parameters more swiftly and can effectively respond to shifts in the data distribution. This heightened adaptability enhances the model's ability to acclimate to new datasets, further affirming the robustness of our approach.

8.3.4 Application of Active Learning in the training labelling

With the setting up of the classifier that effectively uses domain transfer principles across the additive manufacturing image datasets, our research makes progress to extend beyond conventional training methodologies. We carry on an exploration of active learning techniques to further investigate the training and classification performance as them evolves over the query iterations of the active learning approach.

The active learning approach introduced a query strategy to the training of the classification model, enabling it to iteratively improve its performance by strategically selecting and labeling the most informative data samples to be used during the process. This iterative approach allowed us to make efficient use of the labeled data resources and to optimise the performance of the model through an active data selection process. The experiment was conducted through a series of steps, performing a structured and iterative approach. This process involves the following key stages:

1. Active Sample Selection: In the first step, we initiated active sample selection. This involved identifying and selecting specific samples from the pool of unlabeled data that were deemed to be the most informative or uncertain for the current stage of model training;

2. Query for Label: Next, we requested labels for the chosen samples. This entailed seeking human input or supervision to annotate the selected samples with their respective class labels;

3. Train with Queried Sample: The model was updated using the queried samples, which now had their labels. This step enabled the model to learn on the new information and adapt accordingly;

4. Validate for Current Query Iteration: Following the model update, we evaluated its performance on the validation dataset to assess how well it was progressing with the newly acquired labeled samples.

This iterative cycle continued until a human supervisor decided to conclude the training phase. This decision point was typically reached when the validation accuracy achieved a predefined target level, indicating that the model had reached a satisfactory level of performance. In this test, we simply iterate through the pool of training samples in order to fully investigate the trace of the overall learning curve of the classifier.

It is worth noting that for this specific experiment, we adopted a pool-based sampling scenario complemented by the uncertainty sampling query strategy. As documented in [26], this strategy holds an important position as one of the most frequently utilised approaches for initiating generalised sampling within the context of active learning based sample selection. In this case study, it is also particularly applicable on this SLS power bed AM dataset.

The active learning implementation was implemented using Python 3, within the Google Colab environment, providing a robust and accessible platform for our experiments. In the initial stages of the experiment, we initialised the previously mentioned classifier model, with the optimiser of choice being Stochastic Gradient Descent. Our selection of SGD was primarily driven by its ability to deliver stable performance during validation tests, while simultaneously minimizing overfitting concerns. This stability persisted even when the training process was extended well beyond the point of convergence.

It is important to note that although optimisers like Adam and RMSprop demonstrate faster convergence, they often perform larger fluctuations in validation results and minor overfitting issues if the training process continues for an extended duration after reaching convergence. In contrast, while SGD may yield a slightly lower initial result compared to the other optimisers, it displayed untapped potential for further improvement when coupled with active learning techniques. This made it the optimiser of choice for our further experimental investigations.

Throughout the course of this experiment, we trained the classifier using 2,000 initial training samples. The active learning process involved a total of 40 queries, with each query actively selecting 50 samples based on the uncertainty sampling query strategy. This approach enabled us to make strategic choices in selecting informative data points for labeling, thereby optimising the performance of the model iteratively.

Following the selection and querying of these samples, a human annotator assigned labels to them. Subsequently, these newly labeled samples were incorporated into the training dataset to fine-tune the classifier and enhance its performance. After each query iteration, we evaluated the performance of the classifier using classification accuracy on the validation dataset. we present the results obtained on the test dataset in the next section.

8.3.5 Results from Active Learning applied to training and labelling

With the introduction of active learning, the evolution of validation accuracy in each query iteration is visually represented in Figure 8.3. The results shown in this figure emphasised the significant impact of active learning on the performance of model.

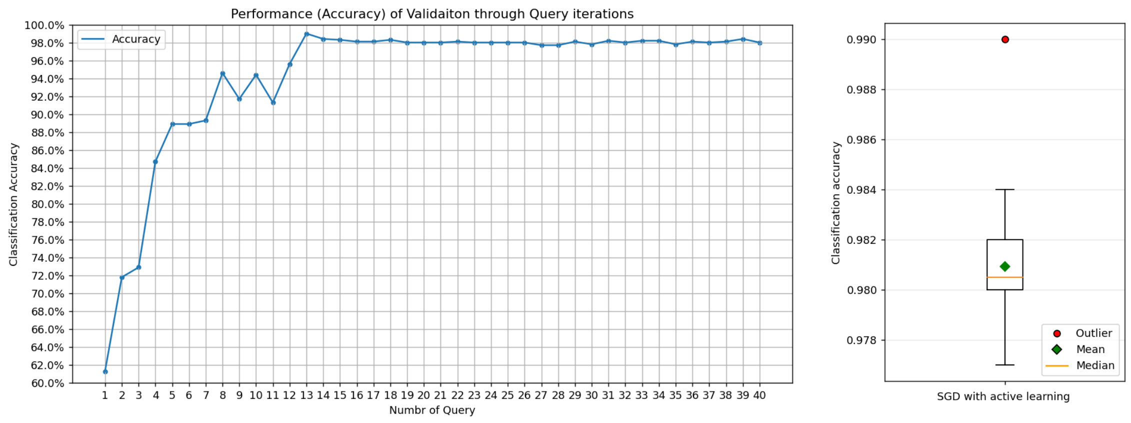

Figure 8.3: Classification accuracy on the validation dataset at the end of each query iteration

The outcomes demonstrate that with the incorporation of active learning, the model achieves convergence after the 13th query, attaining a validation accuracy of approximately 98%. More precisely, the calculated mean validation accuracy from the 13th to the 40th queries stands at 0.981, with a standard deviation of 0.0246 and a peak accuracy of 0.990. Notably, this accuracy level slightly surpasses the result obtained by the SGD optimizer based model without active learning technique applied as presented in Table 8.1, and exhibits a 1% improvement over the baseline. Furthermore, the performance of the classification model remains relatively stable after convergence.

More importantly, the results highlight that the model only requires the first 650 most informative samples to achieve its peak performance, which is merely 32.5% of the total 2,000 labeled training data images. This underscores the efficiency gains achieved through active learning as it enables the model to maximise its performance using a substantially reduced subset of the labeled dataset.

This trained model was then used to classify the labels on the validation dataset mentioned in Section 8.2 in order to evaluate the performance using the relevant metrics the and the results are shown in Table 8.2.

Table 8.2: Predicted results on the validation dataset with Precision, Recall and F1-Score

Confusion matrix	Accuracy	Precision	Recall	F1-Score	ROC-AUC
$\begin{matrix} 487 & 13 \\ 3 & 497 \end{matrix}$	0.988	0.994	0.974	0.984	0.998

8.4 Experiments on Pseudo-Labelling with Active Sample Selection and Human-in-the-loop on an Imbalanced Dataset

In this section, we explain the experimental process encompassing pseudo-labelling augmented by active sample selection and HITL features, specifically focusing on an imbalanced dataset. The sequence commences with the generation of pseudo labels utilising the initial classifier. Subsequently, active sample selection and human correction steps are employed to curtail the count of incorrectly assigned pseudo labels. Following this correction phase, the rectified samples are re-sampled to create a balanced batch, which is then used to further fine-tune the classifier. Throughout this process, we document the relevant outcomes and results. For further clarification, up to this point we have evaluated the initial model quality in the previous sections. The trained classifier has demonstrated the capability to achieve over 98% classification accuracy on the validation dataset (refer to Table 8.2). The current stage corresponds to line 2 in Algorithm 6 from Section 7.3.2. In this step, the trained classifier C_i is given. Further steps will be executed in the next few subsections following Algorithm 6.

8.4.1 Initial test with HITL on the validation dataset

The initial investigation into the performance of HITL serves as a demonstration of the concept and also serves as an example to showcase the impact of HITL on the validation dataset within this specific case study involving SLS power bed images from additive manufacturing.

In this brief test, our approach involves the following steps: initially generating pseudo-labels for the validation dataset using a trained classifier model, then actively querying for the 50 most uncertain samples for human supervision and correction. The steps mentioned corresponds to line 3, line 4 and line 5 in Algorithm 6. Following the human correction of the pseudo labels, we evaluated the overall labeling accuracy to assess the effectiveness of HITL in enhancing the quality of the overall labeling task.

It is important to highlight that due to the stochastic nature of training performance, we conducted multiple runs of the experiments. While all procedures for each replication of the tests remained consistent, we have intentionally chosen a subset of representative tests with relevant results to present in Table 8.3, including the highest, lowest, and median outcomes regarding the number of incorrectly labelled samples as well as the percentage that are covered by the active selection out of the total incorrect pseudo labels. The active query selected incorrect labels will be corrected under the supervision of a human and the new rate of correct labels after the correction, which referred as "corrected accuracy" in the table, can be calculated.

Table 8.3: Performance of active selection and human correction for the initial Pseudo-Labeling

Experiment	Confusion matrix	Accuracy	Precision	Recall	F1-Score	ROC-AUC	N queried	N covered (%)	Corrected accuracy
Highest	$\begin{array}{\|c\|c\|} 486 & 14 \\ \hline 5 & 495 \end{array}$	0.981	0.963	0.98	0.972	0.996	50	**15(78.9%)**	**0.996**
Lowest	$\begin{array}{\|c\|c\|} 487 & 13 \\ \hline 6 & 494 \end{array}$	0.981	0.979	0.979	0.979	0.997	50	**8(40%)**	**0.989**
Median	$\begin{array}{\|c\|c\|} 486 & 14 \\ \hline 8 & 492 \end{array}$	0.978	0.988	0.988	0.988	0.997	50	**14(63.6%)**	**0.992**

As these selected examples are provided solely for the purpose of offering a brief evaluation of active selection with human correction on the pseudo labels, the update of the classifier is not involved in this test. After this test we chose the classifier from the median result as the initial classifier for the subsequent experimental steps. More comprehensive experiments and detailed results will be presented in subsequent sections

8.4.2 Experiments on the imbalanced dataset

This experiment is conducted to evaluate the performance of our approach on an imbalanced dataset. The imbalanced dataset used in this experiment is derived from the testing dataset, which initially consisted of a balanced set of 500 defect samples and 499 normal samples (out of the 500 normal samples, one image is corrupted). For the imbalanced dataset, we randomly selected 101 defect samples from the original testing dataset and combined them with the 499 normal samples, resulting in a new dataset with an imbalanced distribution totaling 600 samples.

The primary steps of this experiment are outlined below, and the corresponding outcomes will be presented for each step:

1. Generate pseudo-label using the trained classifier. This corresponds to line 3 in Algorithm 6, but the dataset tested on is the testing dataset illustrated above rather than the validation data, this differs from Section 8.4.1 ;

2. Active selection according to uncertainty and human correction on the incorrect labeled samples in the selected pseudo-labels. This corresponds to line 4 and 5 in Algorithm 6.

3. Create a new training batch by re-sampling to address the class imbalance issues then update the classifier using the training batch. This corresponds to line 6, 7 and 8 in Algorithm 6

4. Evaluate the performance of the updated classifier on the rest of the pseudo labelled data and the original validation dataset.

8.4.3 Generate pseudo-label using the trained classifier

Referring to step 1 in section 8.4.2, after the initial classification on all the testing data to obtain pseudo labels, the ROC curves and relevant classification report are shown in Figure 8.4 and Table 8.4

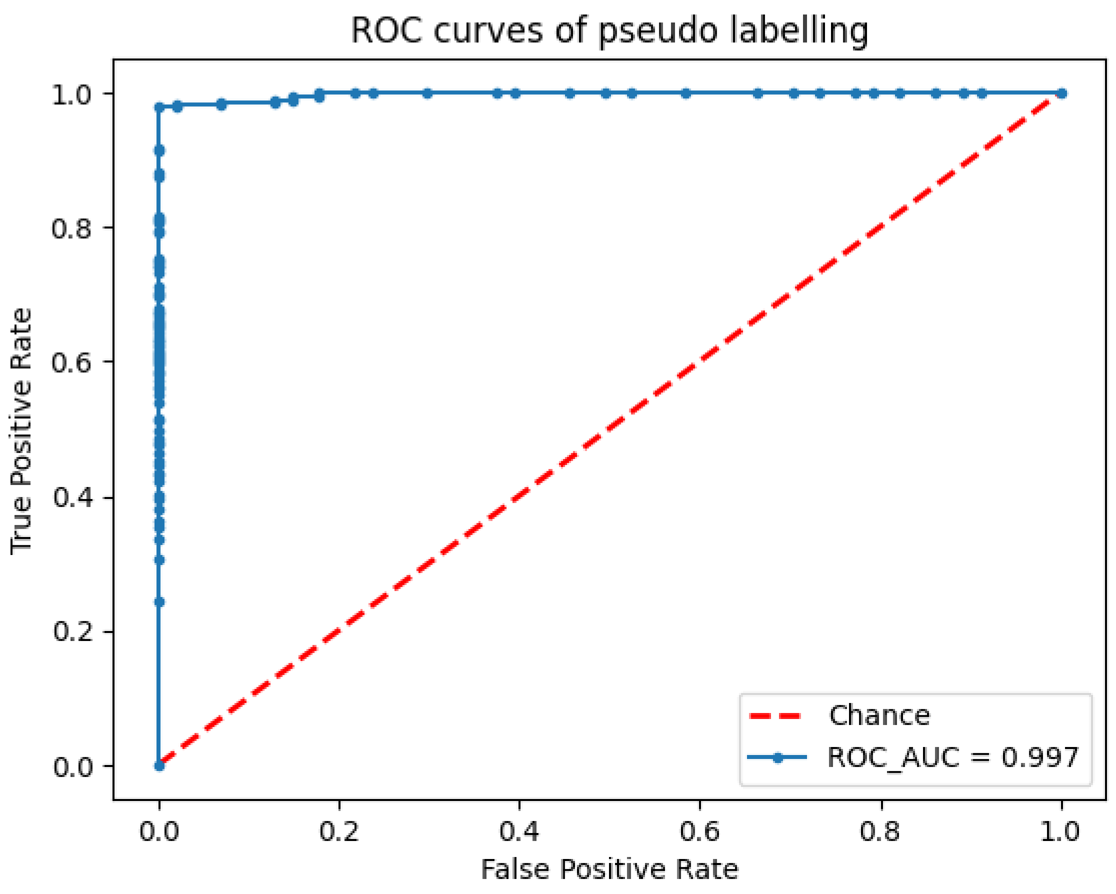

Figure 8.4: ROC curves for classification on the testing dataset for generating the pseudo labels

Table 8.4: Evaluation results of pseudo labelling on the imbalanced dataset

Confusion matrix	Accuracy	Precision	Recall	F1-Score	ROC-AUC
99 | 2 / 9 | 490	0.982	0.917	0.980	0.947	0.997

As shown in the Table 8.4, in the initial classification task to obtain the pseudo labels, from the confusion matrix, there are only 2 samples from the minority class and 9 samples from the majority class, which are the defect and normal class respectively, that are incorrectly labelled yielding 11 mistakes out of the total of 600 pseudo labels. We can consider the results are already high in the performance of this classification task. However, even with such results we will execute further experimental steps to investigate if our HITL approach can further improve the performance of the labelling tasks.

8.4.4 Active selection and human correction on incorrect labels

Referring to step 1 in section 8.4.2 and for further investigation, we conduct active sample selection based on the uncertainty sampling method and queried for 50 samples that are calculated as the most informative for human correction. The results are shown in Figure 8.5 We queried the 50 most uncertain samples and obtained

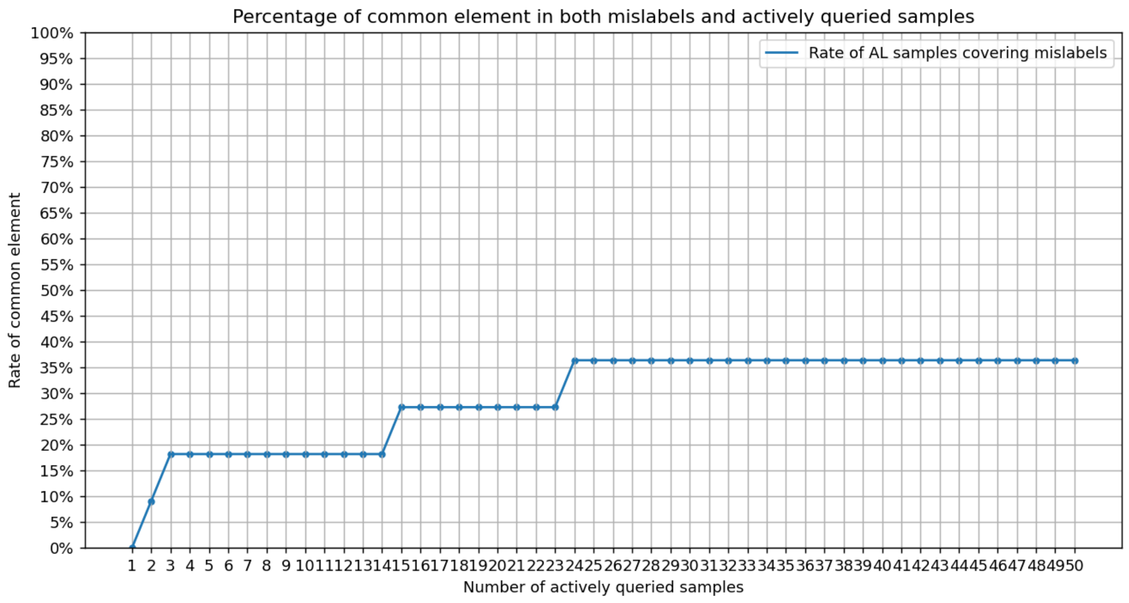

Figure 8.5: Percentage of incorrectly labelled samples that are covered by the number of samples that are active queried

the number of incorrect labels in this 50 samples. It turns out that only 4 out of 11 incorrectly labelled samples are covered by the active query, which yields 36.4% coverage which may seem low. However, if inspect the detail of the 7 incorrectly labelled samples that are not covered by the query, which are shown in Figure 8.6, we find that while all 7 samples are from the normal class, each of them contains features that are relatively close to defect samples and this explains why the classification results are not in agreement with the ground truth but still with high confidence. Some of the images are highly likely to be the defect class even by the justification of a human, so this case is more related to the creation and distribution of the dataset. The 4 corrected labels by the uncertainty sampling and human

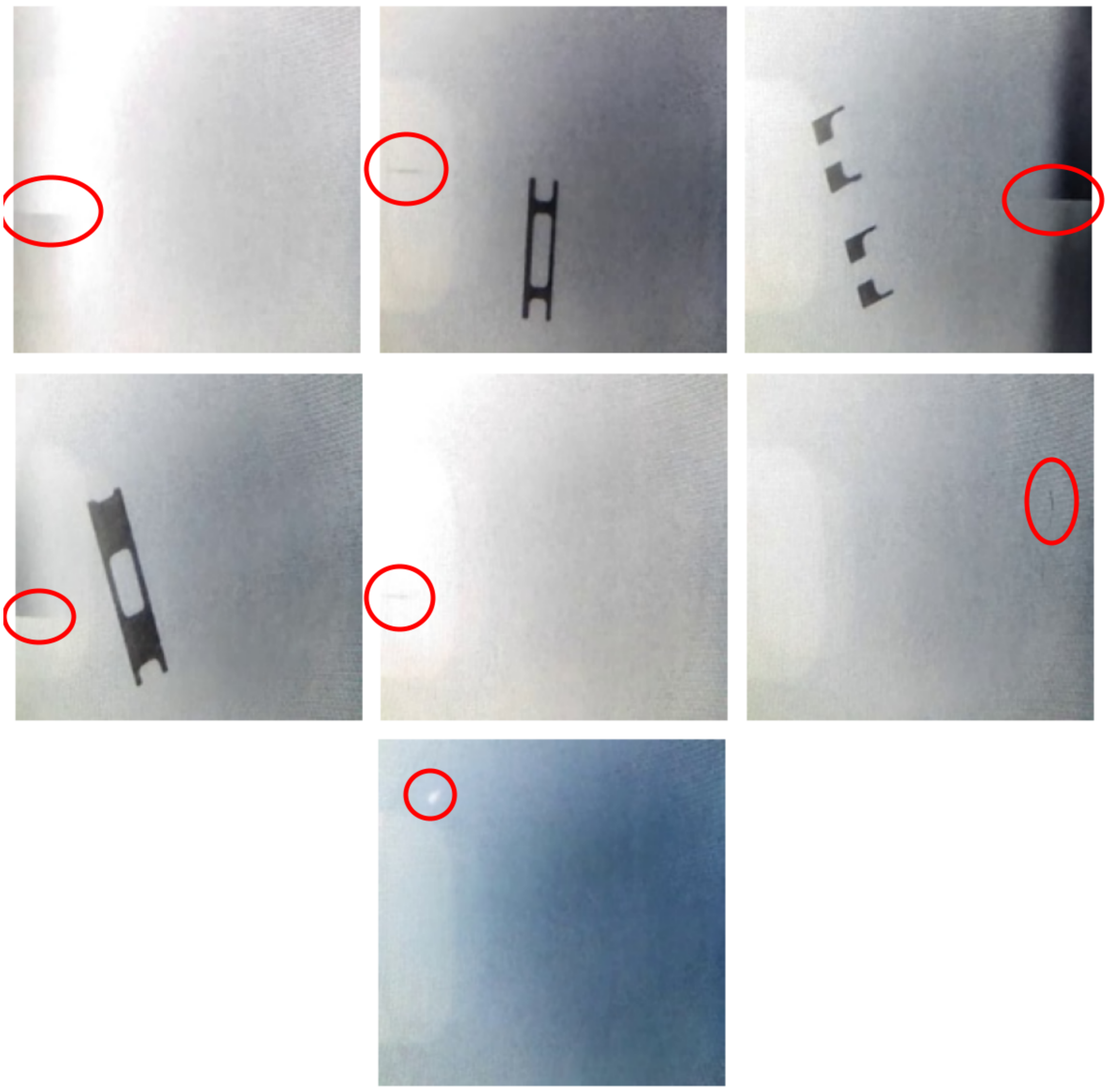
Figure 8.6: The 7 incorrectly labelled samples that not covered by the active query

supervision include 2 samples from the true defect class and 2 from the true normal class. Thus, after human correction, all the defect samples are correctly labelled in this pseudo labelling dataset and the updated confusion matrix and related metrics are shown in Table 8.5

Table 8.5: The updated metrics of the pseudo labelling after human correction

Confusion matrix	Accuracy	Precision	Recall	F1-Score	ROC-AUC
$\begin{array}{\|c\|c\|} 101 & 0 \\ 7 & 490 \end{array}$	0.988	0.935	1.000	0.967	0.997

8.4.5 Create a new training batch by re-sampling

Following step 3 in section 8.4.2, to address the class imbalance issue within the selected 50 samples, we conducted a examination of the class distribution of the samples which revealed that 10 samples belong to the minority class (defect), while the remaining 40 samples were from the majority class (normal). According to Algorithm 5, in this situation, $n_{minot} = 10$ and $n_{major} = 40$, the distribution falls into the envelope where $2 \times n_{minor} < n_{major}$ and the re-sampling process will follow line 15 to 19 in Algorithm 5. All the minority samples will be oversampled by augmentation and $2 \times n_{minor}$, which is 20 in this situation, actively selected samples from the majority class will be used together to form the balanced batch.

As illustrated, to achieve a balance between the minority and majority classes, we applied oversampling by augmenting the 10 minority class samples. Specifically, horizontal flipping was applied for augmentation, as it aligns more naturally with the orientation of the images. Following this oversampling process, the original 10 minority class samples were augmented to a total of 20 samples, effectively increasing the size of the minority class in the dataset. It is worth noting that while oversampling could potentially be repeated to generate more samples, the original testing dataset had already undergone an oversampling process during its initial creation. To avoid excessive duplication of samples and maintain dataset integrity, we decided to apply oversampling only once per sample. Furthermore, we implemented undersampling by selecting and retaining only the first 20 most informative samples from the majority class using an uncertainty sampling strategy.

The re-sampling approach that was applied ensures a balanced representation of both classes within the queried samples, effectively addressing the class imbalance issue. As a result, a new balanced training batch is formed by merging the oversampled minority class samples with the undersampled majority class samples. This balanced training batch comprises a total of 40 samples, with 20 samples from each class. This balanced training batch is then used to update the classifier model, enhancing its ability to handle class imbalance effectively. In cases where the number of samples from the minority class is significantly lower than the majority class, additional queries can be initiated to collect more minority samples until there are sufficient samples to create a training batch.

8.4.6 Re-testing on the rest of the pseudo labelled data and the original validation dataset

To evaluate the the performance of the updated classifier, which is step 4 in Section 8.4.2, with the newly formed training batch and the assumption that all human-labeled samples accurately align with the ground truth, we proceed to fine-tune the classifier using this training batch for 30 epochs. This fine-tuning process is conducted in a manner similar to the active learning-based training applied during the initial training of the classifier.

Following this fine-tuning step, the updated classifier is then evaluated using 2 datasets. One is the original validation dataset, the other is the remaining 550 pseudo samples which are derived from the total of 600 testing samples with the exclusion of the 50 samples that were previously selected through active queries.

Following the classifier update using the training batch, the outcomes for the remaining pseudo labels remain consistent with their initial classifications. This observation aligns with our expectations, given that the seven erroneous labels exhibit high confidence. Moreover, the new training batch primarily emphasises samples that induce substantial uncertainty in the classifier. It is noteworthy that the seven incorrectly labeled samples, as elaborated in Figure 8.6 within Section 8.4.4, are characterised by features strongly indicative of classification into the opposite class.

Conversely, the updated the performance of the classifier on the validation dataset

Table 8.6: Results of classification on the rest of the pseudo labelled data and the original validation dataset

Dataset	Classifier	Confusion matrix	Accuracy	Precision	Recall (TPR)	F1-Score	ROC -AUC
Remained pseudo labels	Initial	$\begin{array}{c\|c} 91 & 0 \\ \hline 7 & 452 \end{array}$	0.989	0.929	1.000	0.963	0.999
	Updated	$\begin{array}{c\|c} 91 & 0 \\ \hline 7 & 452 \end{array}$	0.989	0.929	1.000	0.963	0.999
Validation dataset	Initial	$\begin{array}{c\|c} 486 & 14 \\ \hline 8 & 492 \end{array}$	0.978	0.984	0.972	0.978	0.993
	Updated	$\begin{array}{c\|c} \mathbf{493} & \mathbf{7} \\ \hline \mathbf{4} & \mathbf{496} \end{array}$	**0.989**	**0.992**	**0.986**	**0.989**	**0.998**

demonstrates significant enhancement. It reduces the misclassification of defect samples from 14 to 7 and normal samples from 8 to 4, consequently increasing the overall accuracy from 0.978 to 0.989. While the absolute improvements may appear small, it is important to recognise that the aim of our approach is to enhance classifier performance through HITL, even when commencing from an already high level baseline accuracy.

8.4.7 Discussion

In contrast to the work presented by [25], our approach, as outlined in this chapter, offers the following notable contributions in the aspect of defect classification on the image dataset from AM:

1. By leveraging transfer learning and fine-tuning the classification model in our approach shows superior accuracy and outperforms it in various related evaluation metrics;

2. Furthermore, by introducing active learning based sampling strategies into the training process, our approach excels in data efficiency, particularly addressing the "small data challenge" by optimising the utilisation of limited data resources;

3. Additionally, we proposed an alternative approach for a re-sampling technique to solve data imbalance issues effectively;

4. Finally, we have devised a labeling mechanism employing semi-supervised learning techniques and a human-in-the-loop component. This mechanism not only facilitates the correction of mis-labeling but also contributes to the continual improvement of the classifier. This aspect is also particularly significant in the context of the "small data challenge."

While our model enhances classification accuracy, it is worth noting that the absolute improvements we obtained, when compared to the baseline, may appear relatively small due to the fact that the initial accuracies of the baseline are already quite high. Nevertheless, our primary objective was to demonstrate how the human-in-the-Loop features can further enhance the performance of the labelling work, even when starting from a high baseline level, which is what we have achieved in this chapter of the thesis.

8.5 Next Steps

In this case study, a series of experiments have been conducted to evaluate the approaches proposed in the previous chapters. This future work section can be used to summarise several possible further investigations that can be continued from the current research outcome:

Exploring different sampling strategies during the active learning-based training process could be a promising direction for future work. While uncertainty sampling has been commonly used, investigating alternative strategies beyond the current uncertainty sampling approach could provide valuable insights. For example, strategies such as diversity sampling or representative sampling could be explored to enhance the efficiency and effectiveness of the active learning process. Additionally, if more suitable datasets become available, further experimentation with various sampling strategies could offer new perspectives on how to select informative samples for model training.

The current stopping criteria for training the classifier in this case study are based on the classification performance on the validation dataset. Exploring the impact of different stopping criteria is a direction for future investigation. Additionally, we have considered the challenges and opportunities that arise when incorporating more data samples from manufacturing, particularly in updating the validation set through further leveraging the active learning strategy. This aspect can also be considered as another direction for future work and development.

In this chapter, we have completed a full cycle of training and labeling work. Since all the positive samples were correctly classified within the remaining testing samples, we have exhausted the defect samples in this testing collection. In the future, if a suitable dataset with a larger sampling size of defect samples becomes available, further experiments can be conducted to investigate performance through multiple working cycles.

8.6 Chapter Summary

The case study presented in this Chapter has introduced our approach, which involves transfer learning, active learning, and Pseudo-Labelling in conjunction with human-in-the-loop features. The practical implementation of these methodologies on the dataset has been explained in detail, and the associated experiments have been carefully conducted. The procedural details of these experiments along with the relevant results have been presented and documented. Furthermore, we have declared the contributions of our work in comparison to the research presented by [25].

In the domain of additive manufacturing, materials and manufacturing devices come at a high cost and the printing process itself can be time-consuming. The generation of defect samples, whether through artificial methods or capturing naturally occurring defects, leads to a significant financial cost. Moreover, the identification of randomly generated natural defects among a large pool of normal samples is also a challenging task. Consequently, the concept of data efficiency becomes extremely important, as it often encounters the "small data challenge" due to the shortage of properly labelled samples, particularly in the case of defect samples. The approach presented in this chapter represents a valuable contribution aimed at overcoming these challenges and addressing the associated issues within the AM domain.

Chapter 9

Conclusions and Future Work

9.1 Overview

This chapter serves as the conclusion of the thesis, providing an examination of each area investigated throughout the research. Additionally, this conclusions chapter offers a restatement and gathering together as well as some further elaboration on individual conclusions previously discussed in earlier chapters. This enables more overview discussion and analysis of the outcomes derived from the research.[151]

In previous chapters, the central hypothesis of the thesis was subdivided into smaller sections that were tackled individually. In this study, our research began with a deep dive into the challenges posed by the limited availability of the current databases in the field of defect detection during the process of Additive Manufacturing, which we would describe as the small data challenge in AM. To investigate the gap, we delved into the existing open databases and high quality journal articles from the high impact journal *Additive Manufacturing* to conduct a systematic review to highlight the disparities between current AM defect detection datasets and the well-established open-source datasets commonly used in other applications of machine learning and deep learning. It became evident that a significant gap exists between the current AM datasets and standard open-source datasets for other machine learning applications.

After the confirmation of these gaps, we summarised 3 hypotheses and 3 related research questions (RQ), based on which, we developed approaches that leverage the techniques of transfer learning and fine-tuning to enable the training of a classification model with a limited supply of labelled data. This answers the first research question. We then proceeded to involve active learning into the training of the model in order to explore the minimum number of labelled image samples required to achieve convergence during the training process, with a keen focus on optimising data efficiency in order to address the second research question. These approaches along with the SLS Additive Manufacturing defect detection dataset from a systematic review of the literature, enabled a new case study to be completed to further evaluate the approaches we developed and to further improve the labelling mechanism. This was done by leveraging acombination of pseudo-labelling, active sampling and human-in-the-loop features in order to facilitate the correction of mis-labelling and to enhance the performance of the labelling work, even when starting from a high baseline level. Up to this point, our whole working cycle addressed the "small data challenge" in AM and was completed and the last research question was answered.

9.2 Contributions from Corresponding Research Steps

In this section, our objective is to elucidate the contributions we've made throughout the various stages of our research in response to each of the key research questions.

9.2.1 Contribution of the Dataset to the Field of AM

In Chapter 3, we conducted a comprehensive systematic review on the availability of AM datasets. This review proved an important point early in our research showing the lack of readily available open image datasets in the additive manufacturing defect detection domain for machine learning applications, specifically for microstructures in the melt-pools. Our findings emphasised the lack of suitable datasets in AM for computer vision based ML applications, apart from the one we created for our own evaluations as part of this thesis.

The gap in readily available AM datasets for ML defect detection applications justified the creation of our own dataset, which is the emission image dataset obtained from the in-situ monitoring of melt-pools using additively manufactured titanium alloys. This dataset provided essential support during the initial stages of our research. Recognising its value to the broader research community, we took the initiative to publish this dataset as an open-access resource. This contribution of the dataset to the field of additive manufacturing not only supported our own research but also extended its benefits to fellow researchers, facilitating advancements in the domain and at the time of writing it has been downloaded more than 1400 times

9.2.2 Addressing RQ1 and Related Contribution in Defect Classification in AM

RQ1: How can we develop a machine learning model to accurately, with over 95% classification accuracy for example, and efficiently classify the problematic and challenging representation of melt-pool monitoring data even with a limited amount of labeled training data? The proposed approach to the answer involves leveraging transfer learning and fine-tuning methods to develop a classifier capable of addressing this challenge. This methodology is elaborated upon in Chapter 5.

Transfer learning, while a common technique in machine learning, has seen limited exploration within the domain of additive manufacturing. The first research question is "how to create a high accuracy classifier even with very little initial training data". Our approach addressed the challenge of creating accurate classifiers with limited initial training data, specifically targeting the AM domain. We developed an innovative approach that leverages transfer learning and we evaluated its performance on various datasets, including emission images from AM and complex patterns from the DAGM datasets. The results showcased the effectiveness of our approach, particularly on the emission image dataset, where it achieved high accuracy. In a later phase of our research, we conducted a comparative analysis between our model and the approach we developed, with the state-of-the-art work presented by [25], which served as a baseline. Our approach consistently outperformed the baseline best results, even after a subsequent update to the published paper by the authors.

Our CNN based classification model, which incorporates transfer learning using the ImageNet dataset and fine-tuning with target image datasets, has demonstrated its effectiveness in classifying image data from additive manufacturing. The model excels in various classification tasks, achieving high levels of accuracy in the classification tasks between defects and normal patterns. We anticipate that this approach can be extended to address new industrial patterns arising in the context of AM, providing valuable insights and solutions for future challenges in this domain.

While Transfer Learning is a widely employed technique in various machine learning applications, its application within the domain of additive manufacturing related deep learning applications has been relatively scarce. Hence, our research delves into evaluating the efficacy of Transfer Learning in the AM domain, focusing specifically on image datasets derived from AM processes, with the aim of reducing the required number of labeled samples for training the model.

9.2.3 Addressing RQ2 and Related Contribution in AM Tasks

RQ2: Starting with a limited amount of labelled data available, how can we create sufficiently large labeled datasets for training deep learning algorithms without spending a large amount of human resources on the labelling task? The proposed approach to the answer involves applying active learning into the training of the classifier and active sample selection for human annotates to create new labelled data prioritising the most informative samples. The relevant approaches have been illustrated through Chapter 6. The research undertaken in this thesis has addressed the second research question posed in Section 1.3, which pertains to active learning strategies in the context of our approach. Throughout the subsequent chapters, we delved into various active learning scenarios and query strategies, with a specific focus on the uncertainty sampling query strategy. Following a comprehensive assessment of available packages and tools for implementing active learning based algorithms in Chapter 6, we proceeded to develop a practical active learning mechanism based on the acquired knowledge.

This active learning mechanism was devised to complement the fine-tuning process, as outlined in the previous step of our approach. It operates through a continuous cycle of queries initiated by the classifier and responses provided by the annotator, actively contributing to the ongoing enhancement of the overall performance of the model. By prioritising the most informative samples, this method empowers annotators to focus their efforts on labelling samples that significantly contribute to the training process. Simultaneously, the classifier benefits from a stream of more effective training data, resulting in reduced labelling workload and expedited training convergence.

Experiments have been conducted to assess the efficacy of active sample selection strategies in enhancing training outcomes. The results highlight that a higher proportion of actively selected training samples can substantially improve final accuracy compared to randomly selecting training samples. The effectiveness of this approach is further demonstrated in the case study presented in Chapter 8, where it significantly reduces the number of samples required to achieve training convergence by approximately 60%. This reduction is achieved while maintaining a high level of accuracy that competes with the performance obtained when using the entire training dataset. This represents a substantial contribution in addressing the persistent "small data challenge" in additive manufacturing. The use of Active Learning methodologies in labeling tasks is a widespread practice. However, within the domain of AM, such techniques have not been adequately exploited, particularly in the context of image data. Thus, the novelty of this research lies in its application of AL methodologies to AM datasets, where the acquisition of accurate labels is hindered by factors such as time constraints, cost implications, and the inherent limitations of manual annotation processes.

9.2.4 Addressing RQ3 and Related Contribution in Defect classification Practices in AM Data

RQ3: How can we build and evaluate a framework combining transfer learning and fine tuning with active learning for continuously improving the quality of training data and the resulting accuracy of AM defect detection from monitoring images? To answer RQ3, The investigation began by evaluating classifier performance and training approaches using the original datasets (Sections 7.1 and 7.2 in Chapter 7). Subsequently, the study introduced semi-supervised learning with human-in-the-loop for labeling (Section 7.3), alongside a proposed re-sampling approach to address class imbalance issues. These labeling and re-sampling strategies completed the framework refereed in Figure 4.3, and closed the working cycle. After this, to evaluate the developed approaches, a new case study was conducted and presented in Chapter 8. To further evaluate our approach on AM specified data, we conducted a case study on the SLS powder bed image dataset from [25]. Throughout this case study, we developed our labelling mechanism fully. In comparison to the baseline paper, our approach makes several significant contributions to the field of additive manufacturing pattern classification as follows:

Enhanced Accuracy: By utilising transfer learning and fine-tuning, our classification model demonstrates superior accuracy compared to existing methods, as evidenced by various evaluation metrics.

Improved Data Efficiency: We introduce active learning-based sampling strategies into the training process, significantly enhancing data efficiency. This addresses the "small data challenge" by optimising the use of limited data resources.

Novel Resampling Technique: We propose an innovative approach to resampling that effectively tackles the data imbalance issues, ensuring a balanced representation of classes.

HITL Labelling Mechanism: Our labelling mechanism combines semi-supervised learning techniques with human-in-the-loop features. This not only corrects mislabeling but also contributes to ongoing classifier improvement. This aspect is particularly valuable in addressing the “small data challenge” in AM.

In summary, building upon the foundational work of [25], our work has introduced Active Learning and human-in-the-loop features to enhance the original Transfer Learning-based approach. This augmentation has resulted in an improvement in the overall validation accuracy to approximately 98%. Additionally, we have employed resampling methods to construct new training datasets, thereby facilitating further model training.

9.3 Final Thoughts

The research conducted and presented in this thesis has contributed to pave the way for a broader application of transfer learning and fine-tuning to defect detection in AM, with human-in-the-loop as a focal point. To this aim, our framework's capability to generate additional labelled data and our innovative labelling approach are essential contributions. These components not only expedite convergence during the training process while reducing the demand for a large number of training data samples, but also provide a valuable resource for fellow researchers. In fact, we have encountered numerous other researchers facing challenges in initiating their research due to the scarcity of appropriate datasets, and we have witnessed a continuously increasing interest in the dataset we have generated and made available to the research community in October 2022 We aspire that our approaches can contribute to addressing such issues and could support researchers in similar situations by facilitating the generation of high-quality data with minor effort. Finally, we anticipate that our research outcomes can serve as a foundational framework for future researchers within the field of Additive Manufacturing.

Bibliography

[1] Simon Ford, Letizia Mortara, and Tim Minshall. "The Emergence of Additive Manufacturing: Introduction to the Special Issue Technological Forecasting and Social Change". In: *Technological Forecasting and Social Change* 102 (2016), pp. 156–159.

[2] Manisha Bisht et al. "Correlation of selective laser melting-melt pool events with the tensile properties of Ti-6Al-4V ELI processed by laser powder bed fusion". In: *Additive Manufacturing* 22 (2018), pp. 302–306.

[3] Lian Yanping et al. "Fundamental mechanics problems in metal additive manufacturing: A state-of-art review". In: *Advances in Mechanics* 51.3 (2021), pp. 1–54.

[4] Sarah K Everton et al. "Review of in-situ process monitoring and in-situ metrology for metal additive manufacturing". In: *Materials & Design* 95 (2016), pp. 431–445.

[5] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521.7553 (2015), pp. 436–444.

[6] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pp. 248–255.

[7] Li Deng. "The mnist database of handwritten digit images for machine learning research [best of the web]". In: *IEEE Signal Srocessing Sagazine* 29.6 (2012), pp. 141–142.

[8] Tsung-Yi Lin et al. "Microsoft coco: Common objects in context". In: *European Conference on Computer Vision*. Springer. 2014, pp. 740–755.

[9] Franz Pernkopf and Paul O'Leary. "Visual inspection of machined metallic high-precision surfaces". In: *EURASIP Journal on Advances in Signal Processing* 2002.7 (2002), p. 650750.

[10] Xiang Jiang, P Scott, and D Whitehouse. "Wavelets and their applications for surface metrology". In: *CIRP Annals* 57.1 (2008), pp. 555–558.

[11] Fabian Timm and Erhardt Barth. "Non-parametric texture defect detection using Weibull features". In: *Image Processing: Machine Vision Applications IV*. Vol. 7877. International Society for Optics and Photonics. 2011, 78770J.

[12] Ding Shumin, Liu Zhoufeng, and Li Chunlei. "Adaboost learning for fabric defect detection based on hog and SVM". In: *2011 International Conference on Multimedia Technology*. IEEE. 2011, pp. 2903–2906.

[13] Kechen Song and Yunhui Yan. "A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects". In: *Applied Surface Science* 285 (2013), pp. 858–864.

[14] Apostolos Chondronasios, Ivan Popov, and Ivan Jordanov. "Feature selection for surface defect classification of extruded aluminum profiles". In: *The International Journal of Advanced Manufacturing Technology* 83.1-4 (2016), pp. 33–41.

[15] Daniel Soukup and Reinhold Huber-Mörk. "Convolutional neural networks for steel surface defect detection from photometric stereo images". In: *International Symposium on Visual Computing*. Springer. 2014, pp. 668–677.

[16] Daniel Weimer, Bernd Scholz-Reiter, and Moshe Shpitalni. "Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection". In: *CIRP Annals* 65.1 (2016), pp. 417–420.

[17] Priyanka Patel and Amit Thakkar. “The upsurge of deep learning for computer vision applications”. In: *International Journal of Electrical and Computer Engineering* 10.1 (2020), p. 538.

[18] Matthew D Zeiler and Rob Fergus. “Visualizing and understanding convolutional networks”. In: *European Conference on Computer Vision*. Springer. 2014, pp. 818–833.

[19] Maxime Oquab et al. “Learning and transferring mid-level image representations using convolutional neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 1717–1724.

[20] Luiz G Hafemann et al. “Transfer learning between texture classification tasks using convolutional neural networks”. In: *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2015, pp. 1–7.

[21] Jason Yosinski et al. “How transferable are features in deep neural networks?” In: *Advances in Neural Information Processing Systems*. 2014, pp. 3320–3328.

[22] Seunghyeon Kim et al. “Transfer learning for automated optical inspection”. In: *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2017, pp. 2517–2524.

[23] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[24] DAGM Deutsche Arbeitsgemeinschaft für Mustererkennung. *29th Annual Symposium of the German Association for Pattern Recognition Kernel Description*. 2007. URL: `https://conferences.mpi-inf.mpg.de/dagm/2007/index.html` (visited on 08/17/2007).

[25] Erik Westphal and Hermann Seitz. “A machine learning method for defect detection and visualization in selective laser sintering based on convolutional neural networks”. In: *Additive Manufacturing* 41 (2021), p. 101965.

[26] Burr Settles. *Active Learning Literature Survey.* Computer Sciences Technical Report 1648. University of Wisconsin–Madison, 2009.

[27] Zongwei Zhou et al. “Fine-tuning convolutional neural networks for biomedical image analysis: actively and incrementally”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2017, pp. 7340–7351.

[28] Nizar Grira, Michel Crucianu, and Nozha Boujemaa. “Active semi-supervised clustering for image database categorization”. In: *Content-Based Multimedia Indexing* (2005).

[29] Lin Yang et al. “Suggestive annotation: A deep active learning framework for biomedical image segmentation”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention.* Springer. 2017, pp. 399–407.

[30] Fei Gao et al. “A novel active semisupervised convolutional neural network algorithm for SAR image recognition”. In: *Computational Intelligence and Neuroscience* 2017 (2017).

[31] Dino Ienco et al. “Clustering based active learning for evolving data streams”. In: *International Conference on Discovery Science.* Springer. 2013, pp. 79–93.

[32] Dana Angluin. “Queries and concept learning”. In: *Machine Learning* 2.4 (1988), pp. 319–342.

[33] Jia-Jie Zhu and José Bento. “Generative adversarial active learning”. In: *arXiv preprint arXiv:1702.07956* (2017).

[34] Jonathan Zarecki and Shaul Markovitch. “Textual membership queries”. In: *arXiv preprint arXiv:1805.04609* (2018).

[35] Raphael Schumann and Ines Rehbein. “Active learning via membership query synthesis for semi-supervised sentence classification”. In: *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL).* 2019, pp. 472–481.

[36] Eric B Baum and Kenneth Lang. "Query learning can work poorly when a human oracle is used". In: *International Joint Conference on Neural Networks.* Vol. 8. Beijing China. 1992, p. 8.

[37] Ian Goodfellow et al. "Generative adversarial networks". In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[38] Liantao Wang et al. "Active learning via query synthesis and nearest neighbour search". In: *Neurocomputing* 147 (2015), pp. 426–434.

[39] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[40] Arash Mehrjou, Mehran Khodabandeh, and Greg Mori. "Distribution aware active learning". In: *arXiv preprint arXiv:1805.08916* (2018).

[41] Les Atlas, David Cohn, and Richard Ladner. "Training connectionist networks with queries and selective sampling". In: *Advances in Neural Information Processing Systems.* Ed. by D. Touretzky. Vol. 2. Morgan-Kaufmann, 1989.

[42] Vikram Krishnamurthy. "Algorithms for optimal scheduling and management of hidden Markov model sensors". In: *IEEE Transactions on Signal Processing* 50.6 (2002), pp. 1382–1397.

[43] Hwanjo Yu. "SVM selective sampling for ranking with application to data retrieval". In: *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining.* 2005, pp. 354–363.

[44] Sanjoy Dasgupta, Daniel J Hsu, and Claire Monteleoni. "A general agnostic active learning algorithm". In: *Advances in Neural Information Processing Systems* 20 (2007).

[45] Tuan Pham et al. "Stream-based active learning for sliding windows under the influence of verification latency". In: *Machine Learning* 111.6 (2022), pp. 2011–2036.

[46] David D Lewis. “A sequential algorithm for training text classifiers: Corrigendum and additional data”. In: *Acm Sigir Forum.* Vol. 29. 2. ACM New York, NY, USA. 1995, pp. 13–19.

[47] Steven CH Hoi, Rong Jin, and Michael R Lyu. “Large-scale text categorization by batch mode active learning”. In: *Proceedings of the 15th International Conference on World Wide Web.* 2006, pp. 633–642.

[48] Jing Liu et al. “Identifying Adverse Drug Reaction-Related Text from Social Media: A Multi-View Active Learning Approach with Various Document Representations”. In: *Information* 13.4 (2022), p. 189.

[49] Cha Zhang and Tsuhan Chen. “An active learning framework for content-based information retrieval”. In: *IEEE Transactions on Multimedia* 4.2 (2002), pp. 260–268.

[50] Zhikai Yang et al. “Deep Co-Training Active Learning for Mammographic Images Classification”. In: *2020 Chinese Automation Congress (CAC).* 2020, pp. 1059–1062.

[51] Tomás Sabata, Petr Pulc, and Martin Holena. “Semi-supervised and active learning in video scene classification from statistical features.” In: *Ial@ Pkdd/Ecml* 2192 (2018), pp. 24–35.

[52] Wenbin Cai, Ya Zhang, and Jun Zhou. “Maximizing expected model change for active learning in regression”. In: *2013 IEEE 13th International Conference on Data Mining.* IEEE. 2013, pp. 51–60.

[53] Soheil Mohajer, Changho Suh, and Adel Elmahdy. “Active learning for top-K rank aggregation from noisy comparisons”. In: *International Conference on Machine Learning.* PMLR. 2017, pp. 2488–2497.

[54] Manali Sharma and Mustafa Bilgic. “Evidence-based uncertainty sampling for active learning”. In: *Data Mining and Knowledge Discovery* 31.1 (2017), pp. 164–202.

[55] Wenting Dai et al. “Towards Automatic Optical Inspection of Soldering Defects”. In: *2018 International Conference on Cyberworlds (CW)*. IEEE. 2018, pp. 375–382.

[56] H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. “Query by committee”. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. 1992, pp. 287–294.

[57] Ido Dagan and Sean P Engelson. “Committee-based sampling for training probabilistic classifiers”. In: *Machine Learning Proceedings 1995*. Elsevier, 1995, pp. 150–157.

[58] Andrew McCallum, Kamal Nigam, et al. “Employing EM and Pool-Based Active Learning for Text Classification.” In: *ICML*. Vol. 98. Madison. 1998, pp. 350–358.

[59] Burr Settles, Mark Craven, and Soumya Ray. “Multiple-instance active learning”. In: *Advances in Neural Information Processing Systems* 20 (2007).

[60] Nicholas Roy and Andrew McCallum. “Toward optimal active learning through monte carlo estimation of error reduction”. In: *ICML, Williamstown* 2 (2001), pp. 441–448.

[61] David Cohn. “Neural network exploration using optimal experiment design”. In: *Advances in Neural Information Processing Systems* 6 (1993).

[62] H Scudder. “Probability of error of some adaptive pattern-recognition machines”. In: *IEEE Transactions on Information Theory* 11.3 (1965), pp. 363–371.

[63] Avrim Blum and Tom Mitchell. “Combining labeled and unlabeled data with co-training”. In: *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*. 1998, pp. 92–100.

[64] Kamal Nigam et al. “Text classification from labeled and unlabeled documents using EM”. In: *Machine Learning* 39.2-3 (2000), pp. 103–134.

[65] Isaac Triguero et al. “On the characterization of noise filters for self-training semi-supervised in nearest neighbor classification”. In: *Neurocomputing* 132 (2014), pp. 30–41.

[66] Ming Li and Zhi-Hua Zhou. “SETRED: Self-training with editing”. In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer. 2005, pp. 611–621.

[67] Yu Wang et al. “Semi-supervised learning based on nearest neighbor rule and cut edges”. In: *Knowledge-Based Systems* 23.6 (2010), pp. 547–554.

[68] N Piroonsup and Sukree Sinthupinyo. “Analysis of training data using clustering to improve semi-supervised self-training”. In: *Knowledge-Based Systems* 143 (2018), pp. 65–80.

[69] Jesper E Van Engelen and Holger H Hoos. “A survey on semi-supervised learning”. In: *Machine Learning* 109.2 (2020), pp. 373–440.

[70] Dong-Hyun Lee et al. “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks”. In: *Workshop on Challenges in Representation Learning, ICML*. Vol. 3. 2. Atlanta. 2013, p. 896.

[71] Minh Thanh Vo et al. “Dealing with the class imbalance problem in the detection of fake job descriptions”. In: *Computers, Materials & Continua* 68.1 (2021), pp. 521–535.

[72] Pallavi Kulkarni and Roshani Ade. “Logistic regression learning model for handling concept drift with unbalanced data in credit card fraud detection system”. In: *Proceedings of the Second International Conference on Computer and Communication Technologies: IC3T 2015, Volume 2*. Springer. 2016, pp. 681–689.

[73] Amira Kamil Ibrahim Hassan and Ajith Abraham. “Modeling insurance fraud detection using imbalanced data classification”. In: *Advances in Nature and Biologically Inspired Computing: Proceedings of the 7th World Congress on*

Nature and Biologically Inspired Computing (NaBIC2015) in Pietermaritzburg, South Africa, held December 01-03, 2015. Springer. 2016, pp. 117–127.

[74] GAOGD Sambasivam and Geoffrey Duncan Opiyo. “A predictive machine learning application in agriculture: Cassava disease detection and classification with imbalanced dataset using convolutional neural networks”. In: *Egyptian Informatics Journal* 22.1 (2021), pp. 27–34.

[75] Gianluigi Folino, Francesco Sergio Pisani, and Pietro Sabatino. “An incremental ensemble evolved by using genetic programming to efficiently detect drifts in cyber security datasets”. In: *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*. 2016, pp. 1103–1110.

[76] Mayank Taneja et al. “Prediction of click frauds in mobile advertising”. In: *2015 Eighth International Conference on Contemporary Computing (IC3)*. IEEE. 2015, pp. 162–166.

[77] Sun Choi et al. “Prediction of weather-induced airline delays based on machine learning algorithms”. In: *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*. IEEE. 2016, pp. 1–6.

[78] Justin M Johnson and Taghi M Khoshgoftaar. “Survey on deep learning with class imbalance”. In: *Journal of Big Data* 6.1 (2019), pp. 1–54.

[79] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. “A systematic study of the class imbalance problem in convolutional neural networks”. In: *Neural Networks* 106 (2018), pp. 249–259.

[80] Guo Haixiang et al. “Learning from class-imbalanced data: Review of methods and applications”. In: *Expert Systems with Applications* 73 (2017), pp. 220–239.

[81] Nitesh V Chawla et al. “SMOTE: synthetic minority over-sampling technique”. In: *Journal of Artificial Intelligence Research* 16 (2002), pp. 321–357.

[82] David G Lowe. “Distinctive image features from scale-invariant keypoints”. In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110.

[83] Yao Lu, An-An Liu, and Yu-Ting Su. “Chapter 6 - Mitosis detection in biomedical images”. In: *Computer Vision for Microscopy Image Analysis.* Ed. by Mei Chen. Computer Vision and Pattern Recognition. Academic Press, 2021, pp. 131–157.

[84] Yang Song and Weidong Cai. “Chapter 4 - Visual feature representation in microscopy image classification”. In: *Computer Vision for Microscopy Image Analysis.* Ed. by Mei Chen. Computer Vision and Pattern Recognition. Academic Press, 2021, pp. 73–100.

[85] Peter Bajcsy et al. “Chapter 7 - Object measurements from 2D microscopy images”. In: *Computer Vision for Microscopy Image Analysis.* Ed. by Mei Chen. Computer Vision and Pattern Recognition. Academic Press, 2021, pp. 159–183.

[86] Harry Wechsler and Fayin Li. “Chapter 10 - Biometrics and robust face recognition”. In: *Conformal Prediction for Reliable Machine Learning.* Ed. by Vineeth N. Balasubramanian, Shen-Shyang Ho, and Vladimir Vovk. Boston: Morgan Kaufmann, 2014, pp. 189–215.

[87] Pinar Muyan-Özçelik et al. “Chapter 32 - Real-time speed-limit-sign recognition on an embedded system using a GPU”. In: *GPU Computing Gems Emerald Edition.* Ed. by Wen-mei W. Hwu. Applications of GPU Computing Series. Boston: Morgan Kaufmann, 2011, pp. 497–515.

[88] Ross Girshick et al. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2014, pp. 580–587.

[89] Chunhui Gu et al. “Recognition using regions”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE. 2009, pp. 1030–1037.

[90] Ross Girshick. "Fast R-CNN". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 1440–1448.

[91] Shaoqing Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks". In: *Advances in Neural Information Processing Systems* 28 (2015).

[92] Kaiming He et al. "Mask R-CNN". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2961–2969.

[93] Shajib Ghosh, Mukhil Azhagan Mallaiyan Sathiaseelan, and Navid Asadizanjani. "Deep Learning-Based Approaches for Text Recognition in PCB Optical Inspection: A Survey". In: *2021 IEEE Physical Assurance and Inspection of Electronics (PAINE)*. 2021, pp. 1–8. DOI: 10.1109/PAINE54418.2021.9707712.

[94] Shajib Ghosh et al. "An End-to-End Marking Recognition System for PCB Optical Inspection". In: *2022 IEEE Physical Assurance and Inspection of Electronics (PAINE)*. 2022, pp. 1–8. DOI: 10.1109/PAINE56030.2022.10014764.

[95] Zhigang Ling et al. "Deep Siamese Semantic Segmentation Network for PCB Welding Defect Detection". In: *IEEE Transactions on Instrumentation and Measurement* 71 (2022), pp. 1–11. DOI: 10.1109/TIM.2022.3154814.

[96] Takeshi Nakazawa and Deepak V. Kulkarni. "Anomaly Detection and Segmentation for Wafer Defect Patterns Using Deep Convolutional Encoder–Decoder Neural Network Architectures in Semiconductor Manufacturing". In: *IEEE Transactions on Semiconductor Manufacturing* 32.2 (2019), pp. 250–256. DOI: 10.1109/TSM.2019.2897690.

[97] Rick W Wright et al. "How to write a systematic review." In: *Clinical Orthopaedics and Related Research (1976-2007)* 455 (2007), pp. 23–29.

[98] Joshua D Harris et al. "How to write a systematic review". In: *The American Journal of Sports Medicine* 42.11 (2014), pp. 2761–2768.

[99] Syrene A Miller and Jane L Forrest. “Enhancing your practice through evidence-based decision making: PICO, learning how to ask good questions”. In: *Journal of Evidence Based Dental Practice* 1.2 (2001), pp. 136–141.

[100] Mark Everingham et al. “The PASCAL visual object classes (VOC) challenge”. In: *International Journal of Computer Vision* 88 (2010), pp. 303–338.

[101] Kenneth Clark et al. “The Cancer Imaging Archive (TCIA): maintaining and operating a public information repository”. In: *Journal of Digital Imaging* 26 (2013), pp. 1045–1057.

[102] Miguel G Lopez et al. “BCDR: a breast cancer digital repository”. In: *15th International Conference on Experimental Mechanics*. Vol. 1215. 2012, pp. 113–120.

[103] Catherine Wah et al. *The Caltech-UCSD birds-200-2011 dataset*. California Institute of Technology, 2011.

[104] Manan Mehta and Chenhui Shao. “Federated learning-based semantic segmentation for pixel-wise defect detection in additive manufacturing”. In: *Journal of Manufacturing Systems* 64 (2022), pp. 197–210.

[105] Xiangli Yang et al. “A survey on deep semi-supervised learning”. In: *IEEE Transactions on Knowledge and Data Engineering* (2022).

[106] Yassine Ouali, Céline Hudelot, and Myriam Tami. “An overview of deep semi-supervised learning”. In: *arXiv preprint arXiv:2006.05278* (2020).

[107] Guo-Jun Qi and Jiebo Luo. “Small data challenges in big data era: A survey of recent progress on unsupervised and semi-supervised methods”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.4 (2020), pp. 2168–2187.

[108] Luke Scime et al. *Layer-wise imaging dataset from powder bed additive manufacturing processes for machine learning applications (Peregrine v2021-03)*. Feb. 2023. DOI: `10.13139/ORNLNCCS/1779073`.

[109] Luke Scime et al. *Layer-wise imaging dataset from powder bed additive manufacturing processes for machine learning applications (Peregrine v2022-10).* Feb. 2023. DOI: 10.13139/ORNLNCCS/1896716.

[110] Luke Scime et al. *Layer-wise imaging dataset from powder bed additive manufacturing processes for machine learning applications (Peregrine v2022-10.1).* Feb. 2023. DOI: 10.13139/ORNLNCCS/1923043.

[111] Javid taheriboroujeni. *Image-based dataset of artifact surfaces fabricated by additive manufacturing with applications in machine learning.* 2021. DOI: 10.17632/ZYZ6CZNM5H.3. URL: https://data.mendeley.com/datasets/zyz6cznm5h/3.

[112] Guoying Dong. *Feature-based CNN network and data for laser powder bed fusion process.* 2021. DOI: 10.17632/8RM9D4YKBT.1. URL: https://data.mendeley.com/datasets/8rm9d4ykbt/1.

[113] Davis McGregor. *Code: Using machine learning to predict dimensions and qualify diverse part designs across multiple additive machines and materials.* 2022. DOI: 10.17632/H8TZPXKVDC.1. URL: https://data.mendeley.com/datasets/h8tzpxkvdc/1.

[114] Douglas Brion and Sebastian Pattinson. *Data set for "Generalisable 3D printing error detection and correction via multi-head neural networks".* 2022. DOI: 10.17863/CAM.84082. URL: https://www.repository.cam.ac.uk/handle/1810/339869.

[115] Xiao Liu. *Emission images normal samples.* Oct. 2022. DOI: 10.6084/m9.figshare.21280173.v1. URL: https://figshare.com/articles/dataset/emission_images_normal_samples/21280173.

[116] Xiao Liu. *Emission images defect samples.* Oct. 2022. DOI: 10.6084/m9.figshare.21280104.v1. URL: https://figshare.com/articles/dataset/emission_images_defect_samples/21280104.

[117] Marc Ackermann. *Metal additive manufacturing porosity images*. Mar. 2023. DOI: 10.6084/m9.figshare.22324993.v1. URL: https://figshare.com/articles/dataset/Metal_additive_manufacturing_porosity_images/22324993.

[118] Ellen Roels et al. *Datasets and images of publication: Additive manufacturing for self-healing soft robots*. Zenodo, Apr. 2020. DOI: 10.5281/zenodo.3757760. URL: https://doi.org/10.5281/zenodo.3757760.

[119] Peter Warren. *ExONE stainless steel 316l Grains 500X*. Dec. 2022. URL: https://www.kaggle.com/datasets/peterwarren/exone-stainless-steel-316l-grains-500x.

[120] R. Wicker and Aaron P. Stebner, eds. *Additive Manufacturing* (2023). URL: https://www.sciencedirect.com/journal/additive-manufacturing.

[121] Xian Yeow Lee et al. "Automated detection of part quality during two-photon lithography via deep learning". In: *Additive Manufacturing* 36 (2020), p. 101444.

[122] Erik Westphal and Hermann Seitz. "Machine learning for the intelligent analysis of 3D printing conditions using environmental sensor data to support quality assurance". In: *Additive Manufacturing* 50 (2022), p. 102535.

[123] Parand Akbari et al. "MeltpoolNet: Melt pool characteristic prediction in Metal Additive Manufacturing using machine learning". In: *Additive Manufacturing* 55 (2022), p. 102817.

[124] Marc Ackermann and Christian Haase. "Machine learning-based identification of interpretable process-structure linkages in metal additive manufacturing". In: *Additive Manufacturing* 71 (2023), p. 103585.

[125] Hyunwoong Ko et al. "Machine learning and knowledge graph based design rule construction for additive manufacturing". In: *Additive Manufacturing* 37 (2021), p. 101620.

[126] Noah H. Paulson et al. "Correlations between thermal history and keyhole porosity in laser powder bed fusion". In: *Additive Manufacturing* 34 (2020), p. 101213.

[127] Odinakachukwu Francis Ogoke et al. "Deep-learned generators of porosity distributions produced during metal Additive Manufacturing". In: *Additive Manufacturing* 60 (2022), p. 103250.

[128] Satyajit Mojumder et al. "Linking process parameters with lack-of-fusion porosity for laser powder bed fusion metal additive manufacturing". In: *Additive Manufacturing* 68 (2023), p. 103500.

[129] Edgar Mendoza Jimenez et al. "Parametric analysis to quantify process input influence on the printed densities of binder jetted alumina ceramics". In: *Additive Manufacturing* 30 (2019), p. 100864.

[130] Ian Goodfellow et al. "Generative adversarial nets". In: *Advances in Neural Information Processing Systems* 27 (2014).

[131] Xin Wang et al. "3D printing of polymer matrix composites: A review and prospective". In: *Composites Part B: Engineering* 110 (2017), pp. 442–458.

[132] Dor Bank, Noam Koenigstein, and Raja Giryes. *Autoencoders*. 2020. DOI: 10.48550/ARXIV.2003.05991. URL: https://arxiv.org/abs/2003.05991.

[133] Fabian Pedregosa et al. "Scikit-learn: Machine learning in python". In: *Journal of Machine Learning Research* 12.85 (2011), pp. 2825–2830.

[134] Piotr Skalski. *Make Sense*. https://github.com/SkalskiP/make-sense/. 2019.

[135] Waleed Abdulla. *Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow*. https://github.com/matterport/Mask_RCNN. 2017.

[136] Tsung-Yi Lin et al. "Feature Pyramid Networks for Object Detection". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 936–944. DOI: 10.1109/CVPR.2017.106.

[137] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2016, pp. 770–778.

[138] Wenyuan Dai et al. "Transferring naive Bayes classifiers for text classification". In: *AAAI.* Vol. 7. 2007, pp. 540–545.

[139] Manali Shaha and Meenakshi Pawar. "Transfer learning for image classification". In: *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA).* IEEE. 2018, pp. 656–660.

[140] Yu-Chuan Su et al. "Transfer learning for video recognition with scarce training data for deep convolutional neural network". In: *arXiv preprint arXiv:1409.4127* (2014).

[141] Ulysse Côté-Allard et al. "Transfer learning for sEMG hand gestures recognition using convolutional neural networks". In: *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC).* IEEE. 2017, pp. 1663–1668.

[142] Allah Bux Sargano et al. "Human action recognition using transfer learning with deep representations". In: *2017 International Joint Conference on Neural Networks (IJCNN).* IEEE. 2017, pp. 463–469.

[143] Francois Chollet et al. *Keras.* 2015. URL: `https://github.com/fchollet/keras`.

[144] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* Software available from tensorflow.org. 2015. URL: `https://www.tensorflow.org/`.

[145] Tivadar Danka and Peter Horvath. "modAL: A modular active learning framework for Python". In: *arXiv preprint arXiv:1805.00979* (2018).

[146] Ying-Peng Tang, Guo-Xiang Li, and Sheng-Jun Huang. *ALiPy: Active Learning in Python.* 2019. DOI: `10.48550/ARXIV.1901.03802`. URL: `https://arxiv.org/abs/1901.03802`.

[147] Yao-Yuan Yang et al. “libact: Pool-based active learning in python”. In: *arXiv preprint arXiv:1710.00379* (2017).

[148] Bill Yuchen Lin et al. “AlpacaTag: An Active Learning-based Crowd Annotation Framework for Sequence Tagging”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations.* Florence, Italy: Association for Computational Linguistics, July 2019, pp. 58–63.

[149] Fabian Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12.85 (2011), pp. 2825–2830.

[150] Ekaba Bisong. “Google Colaboratory”. In: *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners.* Berkeley, CA: Apress, 2019, pp. 59–64.

[151] Xiao Liu, Alessandra Mileo, and Alan F. Smeaton. *A Systematic Review of Available Datasets in Additive Manufacturing.* 2024. arXiv: `2401.15448 [cs.CV]`.