

## A Deep Learning approach to Defect Detection in Additive Manufacturing of Titanium Alloys

X. Liu\*, A. Mileo\*

\*Advanced Manufacturing Research Centre (I-Form), School of Computing, Dublin City University, Dublin, Ireland

### **Abstract**

In Additive Manufacturing (AM) of titanium alloys, the formation of defects in parts is typically related to the stability of the melt pool. With increased instability and size of the melt pool comes an increase in the level of emissions generated as the laser processes the material. Recent developments with in-situ monitoring and process control allows the collection of large amounts of data during the printing process. This includes data about emissions, which are made available as 2D representations in the form of colour images. However, it is still a manual process to inspect these 2D representations to identify defects, which hinders scalability. Given recent advances in Deep Learning for computer vision and the availability of large amounts of data collected from in-situ monitoring, our approach leverages Deep Learning techniques for characterizing abnormal emissions to automatically identify defects during the printing process. One of the challenges to apply deep learning in AM is the lack of proper labelled data for training the models. In this paper, we tackle this challenge by proposing an approach that uses transfer learning and fine-tuning on a pre-trained Convolutional Neural Network (CNN) model called VGG 16 to successfully train the deep model with a small labelled dataset. Results show good classification accuracy on the emission images obtained from the in-situ monitoring system, and improvements in classification of defects on a public industrial benchmark datasets named DAGM (Deutsche Arbeitsgemeinschaft für Mustererkennung e.V., German chapter of the IAPR).

---

### **1 Introduction**

In Additive Manufacturing (AM), also known as 3D printing, a laser scans the input material and a melt pool is created at the laser-material interaction point. This forms a layer of the object being manufactured, and the process continues layer by layer until the manufacture is complete. The formation of defects in 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 leading to the formation of defects in parts. Recent developments in monitoring and process control have resulted in a significant enhancement of the AM process and reduced the amount of inter-build variation and interruption in material manufacturing. Also, 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 is possible to use machine learning on this data to automatically identify key features and predict the presence of defects in manufactured products.

In-situ monitoring facilitates the collection of large amounts of data during the build process in AM. Specifically, this data includes feedback on energy input and emissions from the AM build process, and it is typically done through two sensor modules: one module measures the power of laser input at any point and the other module measures emissions from the melt pool in the near-infrared infrared spectra. The data streams collected from these modules can be used to build 2D and 3D representations of the objects being manufactured. Figure 1 shows one example of the images formed by emissions in 2D and 3D representations produced by the InfiniAM monitoring suite present in the Renishaw machine. Despite the abundance of this type of data, the analysis and characterization of emissions as well as their correlation with defects, is still a manual process that involves examining the representations produced by the monitoring software. As the performance of humans in such tasks can vary, along with fatigue and the associated costs of labor, there is demand 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. Ideally, a real time prediction of a defect during the manufacturing process could tell the operator to either shut down and stop production at an early stage or intervene, if possible, to prevent further defects from happening. This requires not only to develop models with the ability to better and automatically understand the data and relate it to defects, but it also requires to continuously adjust such models as the data is produced. This research aims to explore modern data analytics and machine learning techniques including Convolutional Neural Networks (CNNs) to automatically inspect the huge amounts of data captured by an in-situ monitoring suite to extract the most descriptive features that can help in tasks such as classification and object detection. More specifically, in this research we propose a neural network based architecture to accurately and efficiently identify (and classify) input data as normal or abnormal with a limited amount of labelled initial data.

The material for the 3D-printed parts considered in this paper is Ti6Al4V. The in-situ monitoring suite is the InfiniAM monitoring suite from the Renishaw 3D printer, which can provide feedback on energy input and emissions using two photodiodes as sensors. The sensors detect plasma emissions (range 700 nm to 1040 nm) and near-infrared (range 1090 nm to 1700 nm) from the melt pool and the monitoring suite produces a 3D point cloud model from which we can look 2D representations, which is a horizontal cut of the 3D model. Using a video recording software, we were able to extract videos of the model representing the building process layer by layer. We extracted each of these layers as one image (440px840p resolution) by removing duplicated frames so that we have one image only per layer in the AM process. These images represent the input for our model and from now on we refer to them as to the emission dataset. The rest of the paper is organised as follows: Section 2 is a literature review of related approaches to optical inspection in AM using machine learning; Section 3 illustrates the proposed methodology based on CNN and presents our evaluation setup for two experiments; Section 4 discusses the results obtained from the tests and future research directions; we conclude in Section 6 with a summary and final remarks.

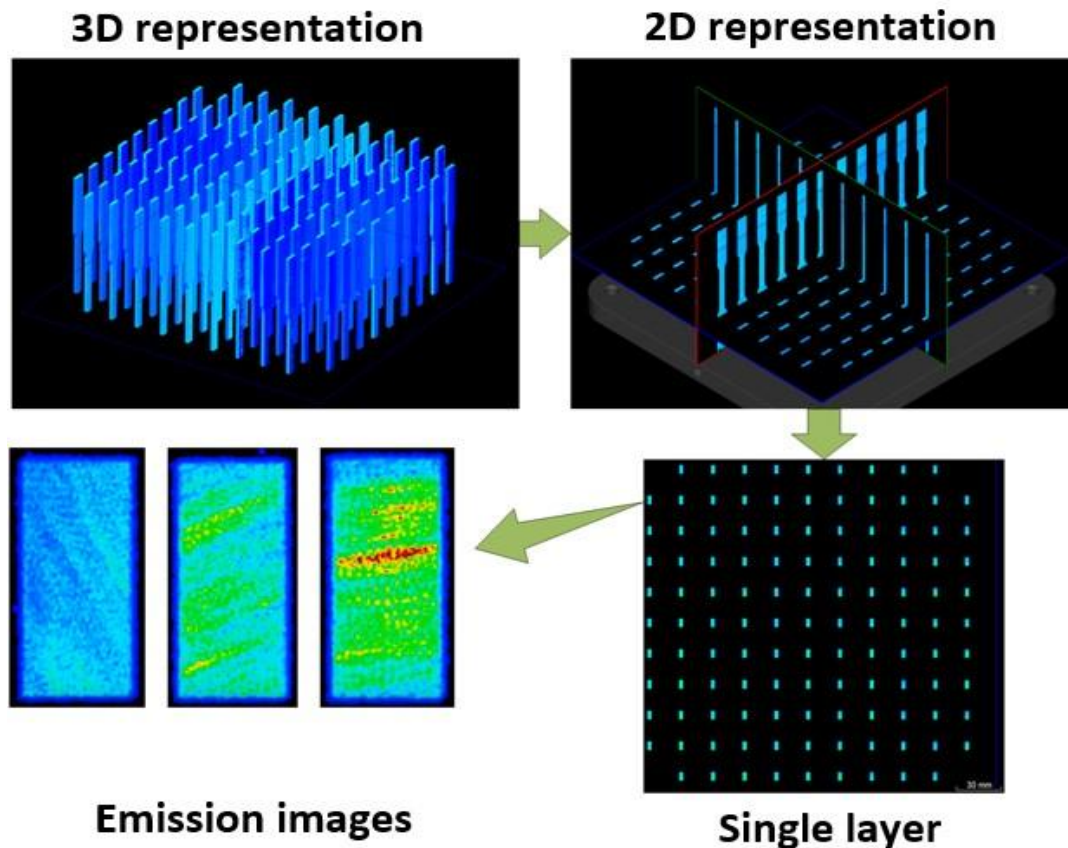


Figure 1: 3D to 2D representations of emissions during manufacture

## **2 Literature review**

During the AM process, as layers are built up to make a component, the height of each layer and the classification of this height is hugely important as the next layer on top of this will be impacted if height is not within certain tolerances. Thus, an optical inspection task is required to detect variations in size, orientation of patterns and other characteristics that could reflect any defects in a layer. It is also important to develop flexible inspection approaches that can be easily reconfigured for different tasks. Many of the conventional approaches to optical inspection are based on feature extraction and further clustering or classification. Over the last decade, advances in deep learning have made it possible to apply Neural Network architectures to automatically inspect the surface of the component and the layers printed during the AM process.

### **2.1 Feature based approaches**

Conventional approaches to AM inspection employ human engineered feature extraction to find defective patterns in an image. The authors of [14] presented a strategy using thresholds derived from a histogram of grey values for segmentation and further computed the shape describing features while in [9] several types of wavelet model have been introduced to extract features from a surface topography. Moreover, [6] proposed an approach for defect detection in texture images with high accuracy 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 such as histogram of oriented gradient (HOG) [15], local binary patterns (LBP) [11] and a gray-level co-occurrence matrix (GLCM) [2]. The results generated by these algorithms for feature extraction and defect detection are high in accuracy. However, their algorithms have limitations in their applicability as they often rely on thresholding. Thresholds are sensitive to background, colour and light. When these conditions change, manual adjustment of the thresholds is needed. Also, such human-engineered methods are limited by the fact that the features need to be customized to specific AM tasks. As a result, for complex AM conditions these methods are not robust and discriminative enough to generate results with sufficient accuracy.

### **2.2 Applying CNNs for inspection in AM**

In order to develop approaches that are both good on adaptability and high in accuracy, several methods based on convolutional neural networks (CNNs) have been proposed. [8] and [5] have applied CNNs in their work and achieved higher classification accuracy than conventional machine learning algorithms. However, a major problem is that training a deep CNN from scratch requires a large amount of labelled data. One of the challenges we consider in this paper is to be able to apply CNNs to AM processes with acceptable accuracy when only a limited amount of labelled data is available.

Authors [1], [13] and [7] have used transfer learning to address this problem by using pre-trained weights from a source network to set the weights of a target network and then use the target network to fulfil the task of feature extraction. However, the performance improvements with these approaches depends on the fact that there is similarity between the source and target domains in their tasks. Authors in [3] have pointed out that if there is a significant difference between the source and target domains such a transfer learning approach with fixed transferred weights can yield less accurate results, and this is the case for AM processes.

### **2.3 Transfer learning with fine-tuning**

In order to address the low performance of transfer learning when using dissimilar target and source domains, authors in [10] proposed a method that applies fine-tuning on the CNN architecture VGG 16 [12]. 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 the data, they used the ImageNet 2012 dataset as the source dataset and the industrial optical inspection dataset DAGM 2007 [16] provided by the German Association for Pattern Recognition, as the target dataset. The results of their experiments showed significant improvements on the overall classification

performance. Building upon this approach, in this paper we develop a methodology to process 2D representations of the emission data from InfiniAM and DAGM for defect detection and analyse changes in performance.

### **3 Methodology**

In this section, the proposed classification method is illustrated and tested using the emission dataset we generated and the DAGM datasets. We first provide a description of the initial dataset and the experimental setup. Secondly, the performance of two different types of classifiers is investigated. Thirdly, the setup of the deep learning model is modified to improve the classification accuracy and all the datasets are processed by the improved model with a comparison of performance. For reproducibility, we also provide a list of specific parameters and settings used in the overall deep learning model and in the training process.

#### **3.1 Initial datasets creation**

The first set of data 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 in a group of dog-bone shaped testing parts of Titanium alloy (Ti6Al4V) during the AM process. 11,000 images were manually exported from the InfiniAM monitoring software as described in the introduction. Initially, all the raw image data are unlabelled. These raw data are not suitable to be used directly for training. For initial inspection, 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 ML 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 sample 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 of 12 classes. Based on the original DAGM dataset, from each pattern, 150 non-defective images samples were selected from the original 1000 non-defective samples to create new groups of testing dataset that involves 6 patterns and each pattern include 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 defected and non-defected classes. Figure 2 shows a collection of examples for normal and defect from each pattern. All the tests in the following sections are based on these datasets.

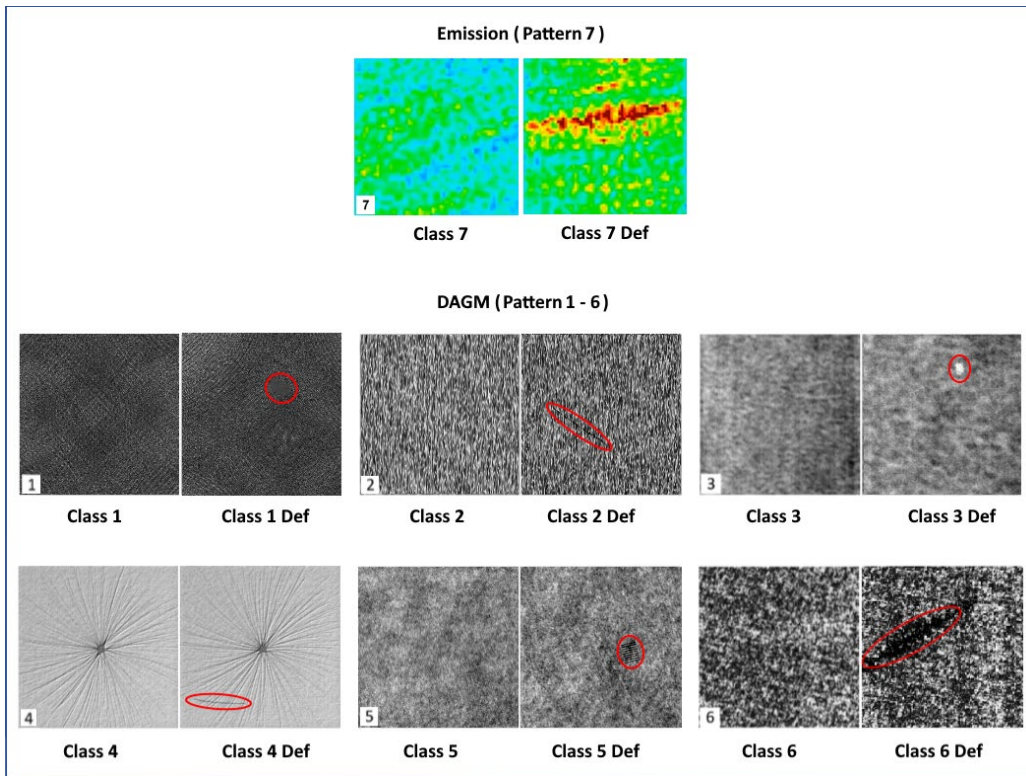


Figure 2: Sample images marked as pattern 1-6 from the DAGM industrial optical inspection dataset. The 6 different texture patterns contain both normal and defects in a particular location, marked with red ellipsoids in the figures. Images marked as pattern 7 are those we generated from the InfiniAM emission dataset.

### 3.2 Combining VGG 16 and SVM classifier

In this test, the image datasets are used as input for features extraction, The features are extracted by the CNN architecture from a VGG16 model using transferred weights trained using ImageNet data [4] without fine-tuning. Then, features are passed to a Support Vector Machine (SVM) classifier to detect if the input image is normal or has a defect. The architecture of this model is shown in Figure 3.

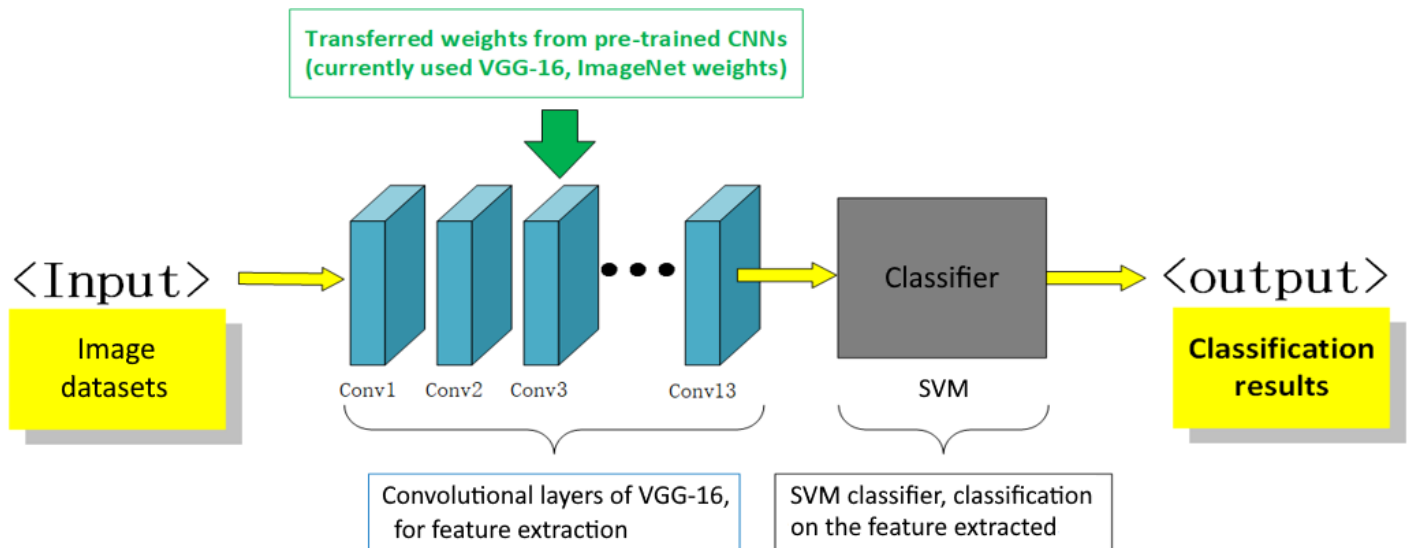


Figure 3: The architecture of the model using a combination of VGG 16 and SVM.

The SVM classifier divides the dataset into classes by creating hyperplanes in multidimensional space. The SVM classifier in this paper is implemented using the C-Support Vector Classification functions from the Python ML package Scikit-learn [18]. The hyperparameters used for the SVM classifier are the type of kernel, the regularization parameter C and the kernel coefficient Gamma. Table I lists the values as used in our experiments, and a brief description of what these parameters are.

Table I: parameters for the SVM classifier

Name of hyperparameter	Type or Value	Description
Kernel	radial basis function (RBF)	The main function of the kernel to transform the given input data into the required form.
Regularization parameter C	1.0	C is the penalty parameter, which is used to maintain regularization. It represents misclassification or error term of the SVM classifier.
Gamma	1/number of features	The kernel coefficient for the RBF, the value of Gamma depends on the number of features from the data

The 7 patterns were individually passed through the model for training and testing purposes. 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. The classification results are shown in Table II.

Table II: classification results from the SVM classifier

Class	Precision	Recall	F1-score	Support
Class1 Def	0.88	0.15	0.25	42
Class1	0.50	0.98	0.66	48
Class2 Def	0.50	0.52	0.51	42
Class2	0.57	0.54	0.55	48
Class3 Def	0.47	1.00	0.64	42
Class3	0.00	0.00	0.00	48
Class4 Def	0.47	1.00	0.64	42
Class4	0.00	0.00	0.00	48
Class5 Def	1.00	0.79	0.88	42
Class5	0.84	1.00	0.91	48
Class6 Def	0.57	0.92	0.72	42
Class6	0.94	0.35	0.52	48
Class7 Def	0.91	0.98	0.94	42
Class7	<b>0.98</b>	<b>0.92</b>	0.95	48

As the size of each labelled dataset is quite small, it is remarkable to see how, even with limited training data, the results of classification for certain classes (such as class 7) have relatively high value in recall with true positive rate (precision) and true negative rate (recall) being 98% and 92% respectively. Despite these positive results, the approach still shows low performance on classes related to patterns 1,2,3 and 4. There are three possible reasons to explain such a 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 improve results, we propose and test a different approach solely based on CNN, which relies on VGG 16 architecture extended and modified for our specific problem.

### 3.3 Extending VGG 16 for classification using dense layers

This approach relies on a transfer learning method in which the 13 convolutional layers from the pre-trained convolutional layers of VGG16 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 4, 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.

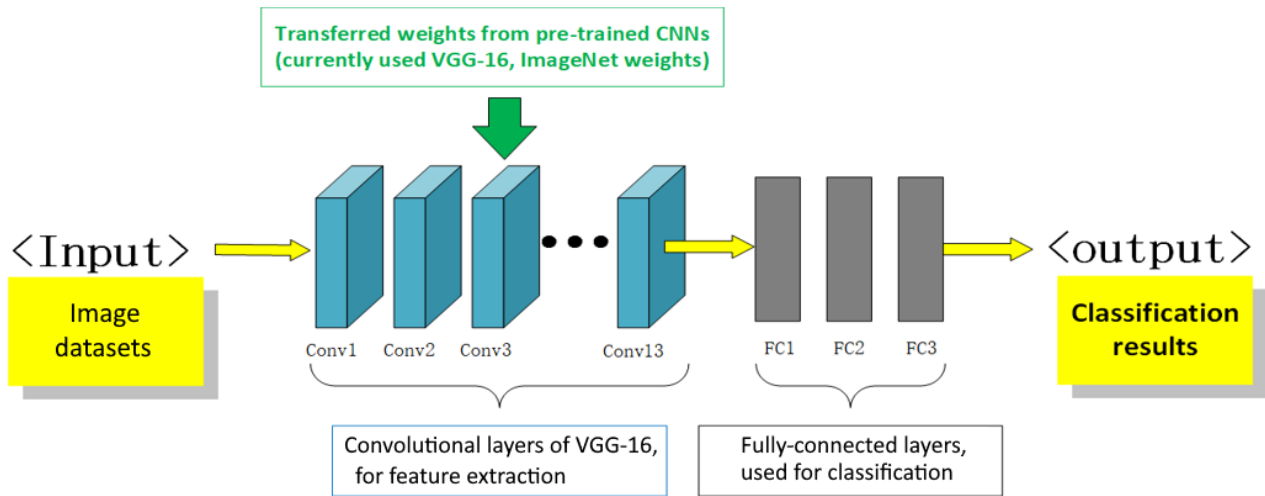


Figure 4: The architecture of the model based on VGG 16 and fully connected layers.

In this experiment, rather than directly test on all the image groups, three patterns which are considered to be the most challenging are selected first. As illustrated in Table II, the selected patterns are: pattern 4, with one of the worst performances in the previous test; pattern 6, with average performance in the previous test; pattern 7, with the best performance in the previous test.

Test on pattern 6: The initial investigation of the performance of the new model began with the dataset of pattern 6 (refer to Figure 2) 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 shows the curves of accuracy and loss for both training and validation for pattern 6.

In Figure 5, 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 the extremely high fluctuation in the validation, the following points are considered:

- 1) The number of samples for the test is relatively small, 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 would be a solution.
- 3) According to Kim et al. (2017) [10] 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 Section 3.4 to reduce the fluctuation and improve the overall classification accuracy.

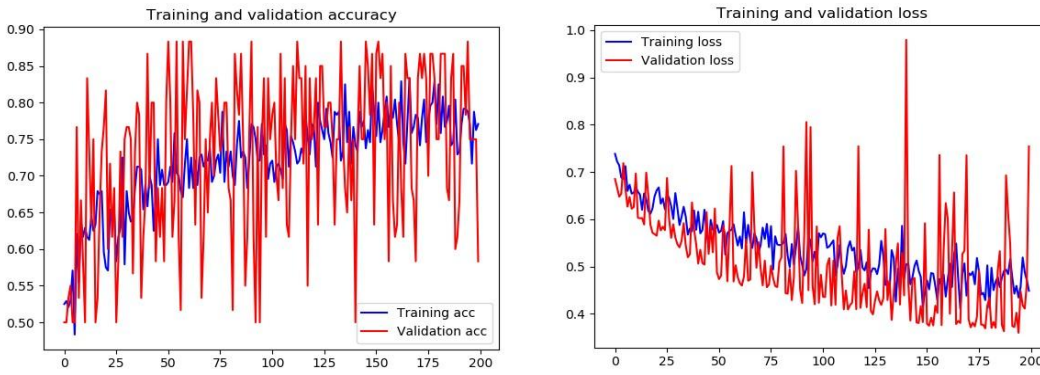


Figure 5: Accuracy and loss over 200 epochs (the horizontal axis shows the number of epochs) with transfer-learning and without fine-tuning. The vertical axis in the left chart stands for the value of accuracy while the vertical axis in the right chart stands for the value of loss.

### 3.4 Further optimisation: pooling and fine-tuning

To further improve the model, an average pooling layer is 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 the 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 6.

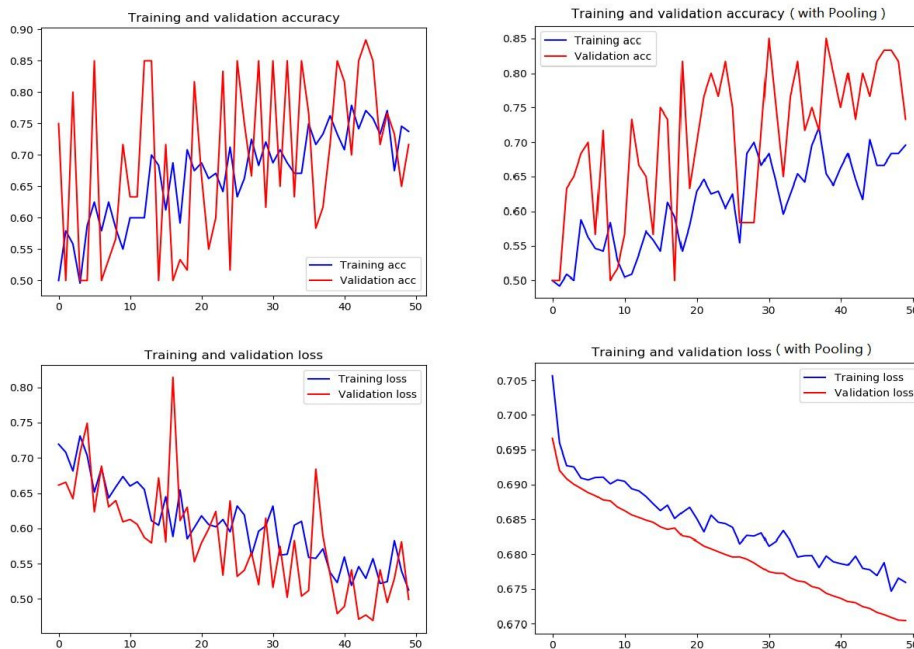


Figure 6: There is an obvious reduction of fluctuation in the curve of the overall accuracy and loss after adding the pooling layers (top-right and bottom-right) compared to the results without pooling a layer (top-left and bottom left).

After improving the model with the addition of a pooling layer, fine-tuning is applied in order to increase the classification accuracy and reduce the loss. Based on this new configuration, the last 3 convolutional layers from the VGG16 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 of training and validation as shown in Figure 7. 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.

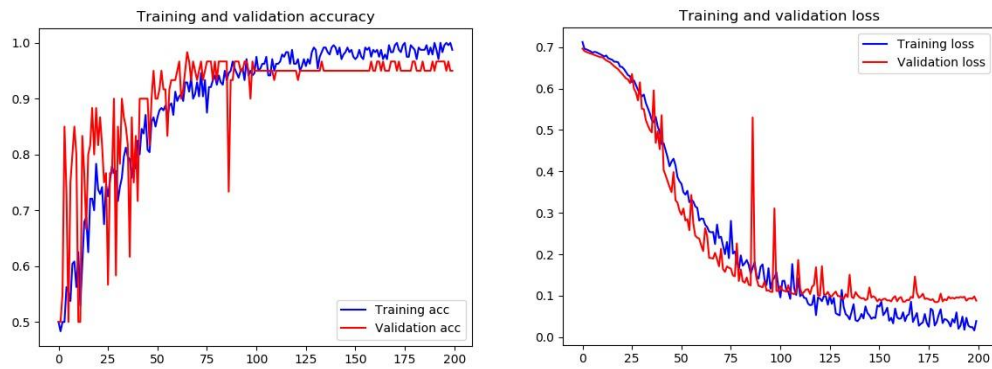


Figure 7: Accuracy and loss for pattern 6, using a model with pooling and fine tuning on the last 3 convolutional layers.

Test on pattern 4: Similar tests are also applied on pattern 4, 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 8). 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.

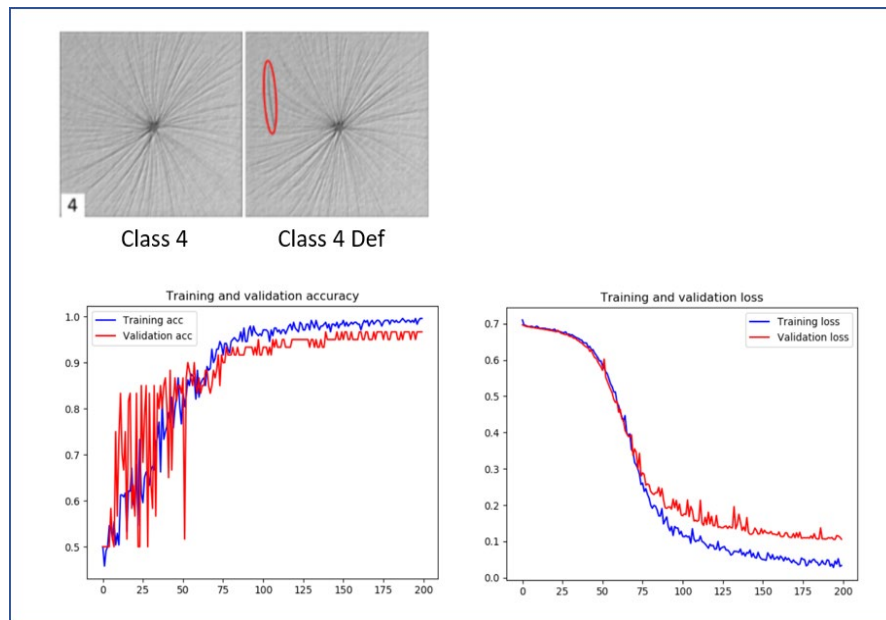


Figure 8: Accuracy and loss for pattern 4, using a model with pooling and fine tuning on the last 3 convolutional layers.

Test on pattern 7: 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 9, the training process reaches convergence after around the 130<sup>th</sup> epoch with a training accuracy around 99% and a validation accuracy over 98%.

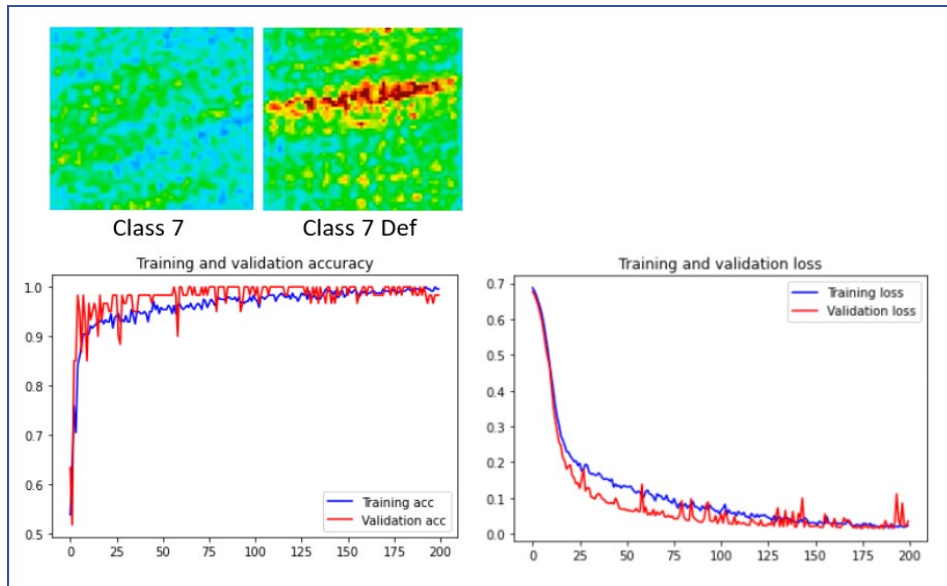


Figure 9: Accuracy and loss for pattern 7, using the model with pooling and fine tuning on the last 3 convolutional layers.

The results of the classification by the model with fine-tuning has shown significant improvement in the overall accuracy. To complete this investigation, the remaining patterns 1, 2, 3 and 5 (refer to Figure 2) are also tested individually using our modified VGG16 architecture with transfer learning and fine-tuning. Results for all the 7 patterns are shown in Table III, 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.

Table III: classification results from the modified VGG16 architecture with transfer learning and fine-tuning

Patterns	Ave. Training Accuracy	Ave. Training Loss	Ave. Validation Accuracy	Ave. 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

### 3.5 Parameters and settings

This subsection specifies the values of parameters used in the overall model, as well as settings for the training process. The deep learning model in this paper is implemented using the Python deep learning package Keras [19]. Figure 10 describes the architecture of the model layer by layer, and the shape of input/output tensors in each layer. Table IV contains the hyperparameters used for the training of the model.

Stopping criterion: In the tests, the numbers of epochs for stopping criterion are simply set as long enough for the curves of validation accuracy and loss to reach the convergence where the curves will not have significant change with longer duration of training. Because the input dataset is different in each test, the number of epochs

required will be different. For example, to reach the convergence in the training process, the test of dataset 1 requires more epochs than that required by the test of dataset 7.

Table IV: hyperparameters used for the training of the deep learning model

Name	Type / Value	Description
Optimizer	Stochastic Gradient Descent (SGD)	Optimizers 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 minimize during training
Learning rate	0.001	The step size at each iteration while moving toward a minimum of a loss function during the training process
Evaluation metric	Accuracy, Loss	Metric is a function to judge the performance of the model

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 is 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 tensor from the VGG 16 model and applies 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 ready to be processed in the dense layers. (See Figure 10)

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.

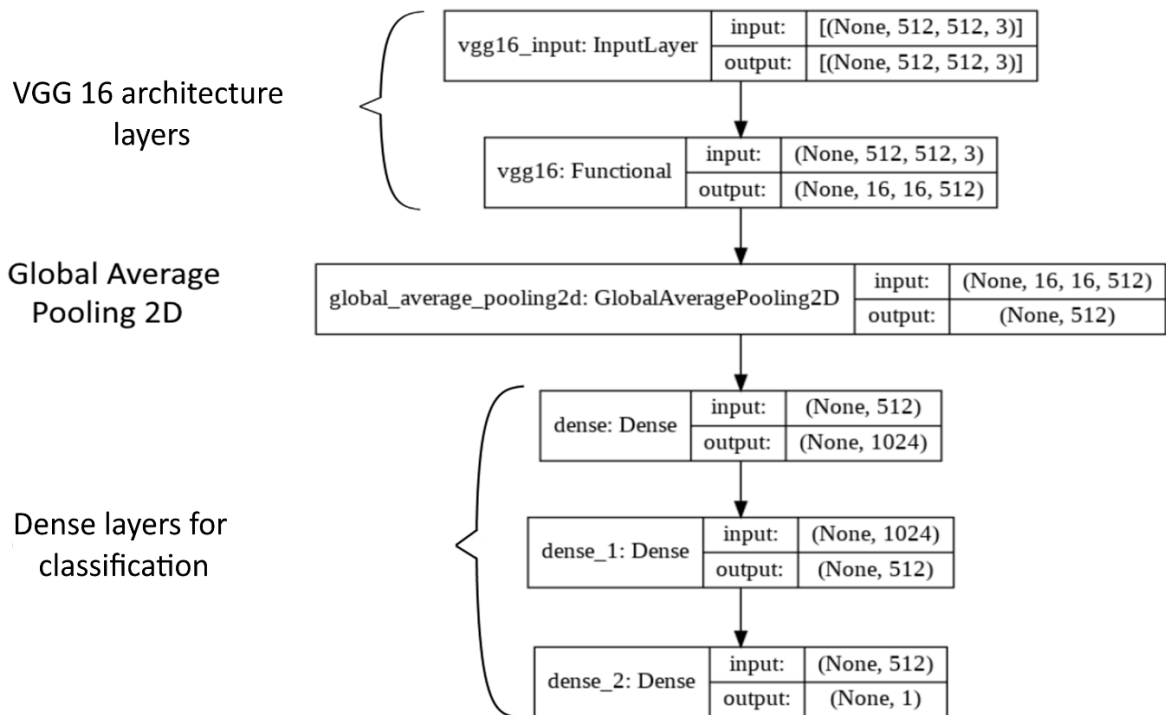


Figure 10: The overall model including the VGG 16 architecture layers, the GAP2D layer and the dense layers for classification. The value “None” in the shape of tensors indicates the dimension is a variable

## **4 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 the 7 patterns in the initial datasets. 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 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 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.

Future research we are planning to investigate goes into two main directions. The first direction is looking to continue to improve the deep learning model. Although the current model works well with the initial datasets, the architecture used in this paper is not the most up to date CNN. Other architectures, such as ResNet [17], have better performance when the problem becomes more complex and the network needs to be deeper. This will be considered for the application of computer vision with transfer learning and fine-tuning on other tasks. The second direction for investigation considers the need to produce more high quality data and we are looking into extending our framework to include active learning and semi-supervised learning for automatic data labelling.

## **5 Conclusion**

In this paper we have designed and tested a deep learning model based on Convolutional Neural Networks to automatically identify defects in the AM process of titanium alloy Ti6Al4V. In our approach, we leverage information about emissions collected through in-situ monitoring and represented as images, one for each layer. These images represent a dataset we have curated and used to train our deep learning model. Our experiments have demonstrated that our model pre-trained and fine-tuned can obtain good performances even with relatively low computational power and limited training data. The model we propose in this paper can be used as an effective feature extractor and classifier with limited labelled data for training. However, 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 limited number of epochs), but it also represents a valuable resource to be used by other researchers. For this reason, we are investigating the combination of our 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. As a result, we will produce a more accurate and more effective pipeline that can potentially be used for different classification tasks where the availability of labelled dataset is scarce. Currently we are looking into two of such tasks: automatic segmentation of individual melt pool areas and characterisation of porosity structure in AM processes.

## **Acknowledgements:**

This research is supported by a research grant from Science Foundation Ireland (SFI) under Grant Number 16/RC/3872 and is co-funded under the European Regional Development Fund and by I-Form industry partners.

## REFERENCES

- [1] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.
- [2] Apostolos Chondronasios, Ivan Popov, and Ivan Jordanov. Feature selection for surface defect classification of extruded aluminum profiles. *The International Journal of Advanced Manufacturing Technology*, 83(14):33–41, 2016.
- [3] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.
- [4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li FeiFei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- [5] Daniel Weimer, Bernd Scholz-Reiter, and Moshe Shpitalni. Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection. *CIRP Annals*, 65(1):417–420, 2016.
- [6] Fabian Timm and Erhardt Barth. Non-parametric texture defect detection using weibull features. In *Image Processing: Machine Vision Applications IV*, volume 7877, page 78770J. International Society for Optics and Photonics, 2011.
- [7] Luiz G Hafemann, Luiz S Oliveira, Paulo R Cavalin, and Robert Sabourin. Transfer learning between texture classification tasks using convolutional neural networks. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2015.
- [8] Daniel Soukup and Reinhold Huber-Mork. Convolutional neural net- works for steel surface defect detection from photometric stereo images. In *International Symposium on Visual Computing*, pages 668–677. Springer, 2014.
- [9] Xiang Jiang, P Scott, and D Whitehouse. Wavelets and their applications for surface metrology. *CIRP annals*, 57(1):555–558, 2008.
- [10] Seunghyeon Kim, Wooyoung Kim, Yung-Kyun Noh, and Frank C Park. Transfer learning for automated optical inspection. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2517–2524. IEEE, 2017.
- [11] Kechen Song and Yunhui Yan. A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects. *Applied Surface Science*, 285:858–864, 2013.
- [12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [13] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724, 2014.
- [14] Franz Pernkopf and Paul O’Leary. Visual inspection of machined metallic high-precision surfaces. *EURASIP Journal on Advances in Signal Processing*, 2002(7):650750, 2002.
- [15] 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*, pages 2903–2906. IEEE, 2011.
- [16] Deutsche Arbeitsgemeinschaft für Mustererkennung e.V., German chapter of the IAPR <https://conferences.mpi-inf.mpg.de/dagm/2007/prizes.html>
- [17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [18] Scikit-learn: machine learning in Python — scikit-learn 0.16.1 <https://scikit-learn.org/>
- [19] Keras: the Python deep learning API <https://keras.io/>