ACS6403 – Final Report

Using Deep Learning to Classify Defects in Laser Metal Deposition – Wire

Group members: Jiannan Tan, Tianhao Zhang

Group No.: GKN Group 2

Submission Date:29th May 2020

Content

1 Table of Figures

2 Table of Tables

3 Introduction

Additive manufacturing is now used by more and more companies for its low cost, high speed, and high precision. It has been widely used in the field of aviation, automobile and other manufacturing fields. However, defects may exist in the manufacturing process. The main purpose of this project is defect detection for laser additive manufacturing (LAM), more specifically: laser additive manufacturing wire-based laser deposition (LMD-w). Our group has been provided with images of defects, and use the method of transfer learning to build a model. Thus, the basic objective of this project is to establish a process monitoring framework for LMD-w.

3.1 Background

3.1.1 Additive Manufacturing

Additive manufacturing has been proved to boost the production rate of manufacturing tremendously. Some redundant parts of the traditional manufacturing process can be omitted. A typical example of AM is 3D printing, which creates objects in reality directly by STL files from CAD software. In the process, in the STL file, the geometric representation of the target is converted into grids, which is digitally separated into discrete two-dimensional layers and sent to the AM system.

The biggest advantage of AM is that it can easily produce objects with complex structures or shapes at one go. The simplified process leads to saving costs and materials. Therefore, AM can bring higher efficiency for some sophisticated areas which are human's future.

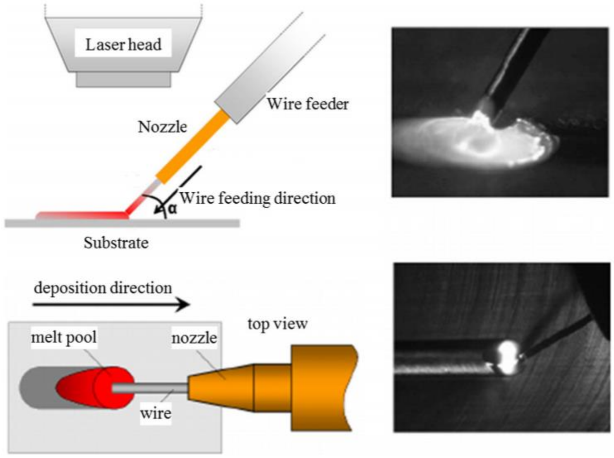

Figure 1-1 LMD-w Process[1]

In order to deal with products of different materials, there are seven AM categories classified by ATSM: Binder jetting, Material jetting, powder bed fusion, directed energy deposition, Sheet lamination, Vat photopolymerization, and Material extrusion. According to the content of the project, the type of processing belongs to directed energy deposition (DED). The process generally involves wire feeding, wire melting, and manipulation over deposition. During processing, the wires would be melted by the laser and deposited on the base. By moving the base or manipulator, the deposits would be stacked on the base like 3d printing.

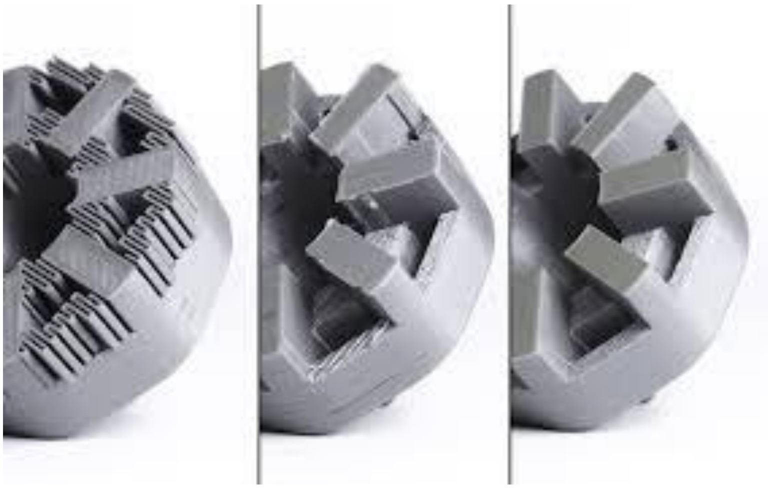

Figure 1-2 Examples of 3-d printing

Otherwise, depending on the type of heat source, the process method also can be divided into three types: laser, arc, and electron beam. The system usually includes a laser, automatic wire feeding system, control system, and auxiliary facilities. Therefore, this project belongs to increasing the recognition of the control system. As the main component, the property of the laser has an important impact on the manufacturing results. Laser power is related to the wire feeding rate, different laser power corresponds to different optimal wire feeding rate, and when the laser power fluctuates, the processing results also change. Therefore, the monitoring system of the deposition process is necessary.

3.1.2 Deep Learning

Deep learning has been widely used for finding the feature representation of massive data. The purpose of using deep learning methods is to build a model to predict the circumstance of the process by camera observed data. The process of modelling is usually referred to as "training", and the output variables used for training are called "training data". By establishing the relationship between the observed value (the input variable) and the output value, the mathematical model can predict the correct result by any new input value to achieve the function of prediction or detection.

Some deep learning problems have thousands of features. For instance, in this project, dozens of different images may represent only one phenomenon, which means each feature has a different weight to the result, and some features would not produce unnecessary deviation to the result. Therefore, these features need to be weighted down to improve training performance. In deep learning, there are two different types of learning: supervised learning and unsupervised learning. The critical difference between the two types is whether the output variables are known. In this project, as the features have been classified, this means it is supervised learning. It would affect the choice of methods.

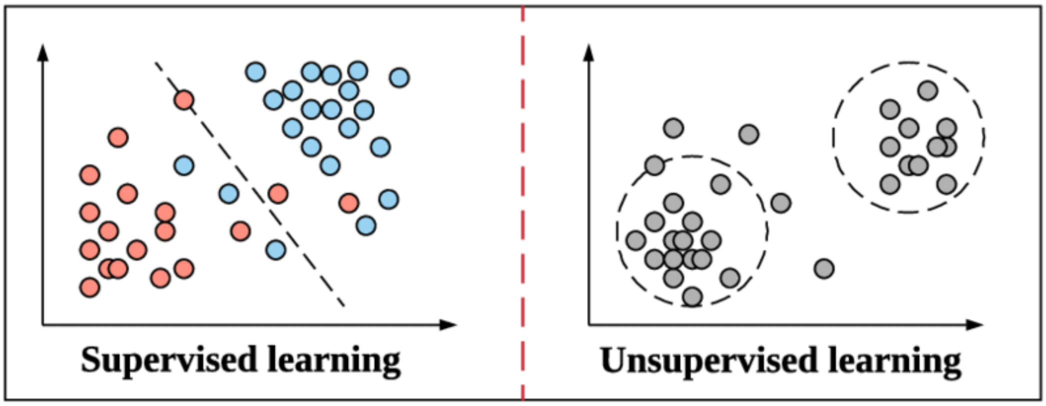

Figure 1-3 Examples of supervised learning and unsupervised learning

Convolutional Neural Network so-called CNN is good at finding the features of different pictures, which means this network could be used to find the features of different defects in the AM process. CNN has an important characteristic and consists of many layers. Each layer can learn different features, as shown in Figure 1-4. The bottom layers can learn some low-level features such as lines and edges. The higher the layers, the more advanced features can be learned, such as the shape of eyes and nose from different people in face recognition. On similar data sets, the features learned at the bottom layers are almost the same, and only the features learned by the several top layers are different. In other words, a lot of manpower and time for collecting massive data would be reduced. The low and midden features learned on the earlier machine are useful for other production machines. Using the features learned by similar datasets for a new dataset is called the transfer learning method. The next paragraph will introduce transfer learning in details.

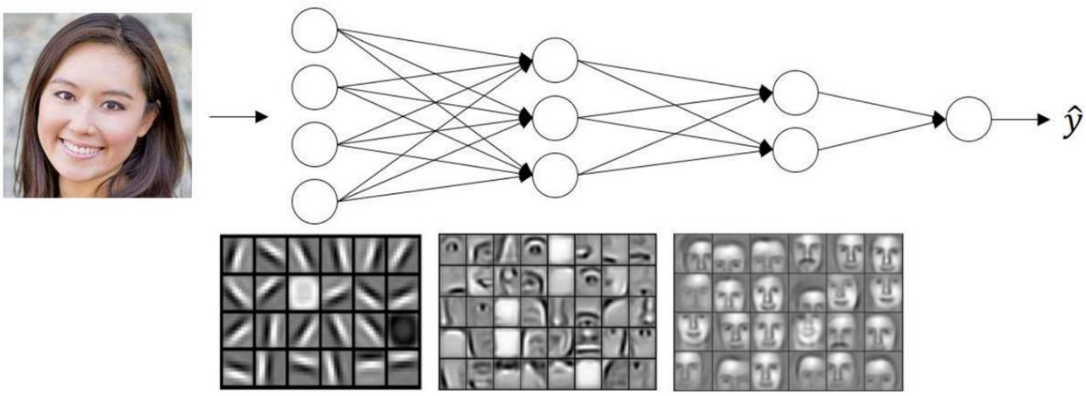

Figure 1-4 Feature representation in different layers of CNN for face recognition [16]

3.1.3 Transfer Learning

Considering the data of project are pictures, and several defects are required to identify, transferring learning is a worthwhile decision. Modelling and training a completely new model for a new defect type can be a huge challenge due to complexity and time limitation. However, there is no need to create a new model in the transfer learning problem. By modifying the parameters in the pre-trained model that has learned the features of some defects, the model can achieve the expected prediction function. That is the main advantage of transfer learning.

To more intuitively introduce transfer learning, Figure 1-5 shows a network structure for a transfer learning case in this project. At first, the pre-trained model MobileNetV2 is trimmed through excluding the classification layers at the top. The reserved layers and parameters from the pre-trained model are named as Base Model. Subsequently, a global average layer is constructed next to the Based Model using a two dimensional Global Average Pooling. The Global Average Pooling could convert the feature map (size: 5x5x1280) learned from Base Model to a vector with 1280 elements. After this layer, a new layer is built for shifting that vector to a score vector of corresponding classes. During the training processes with the AM dataset, all parameters from the Base Model are frozen at the beginning, which means only the parameters in the global average layer and prediction layer are updated. The reason for freezing the Base Model is that the learning ability of the rebuilt network and how the weights learned in the pre-trained model perform should be verified. Then the top few layers in the Base Model are gradually unfrozen, and their parameters are updated in training until the accuracy satisfies the AM requirements. Because the Base Model is a deep CNN, the first few layers always learn very simple and generic features that generalize to almost all types of images, hence, it is unnecessary to spend computational resources and time in those layers.

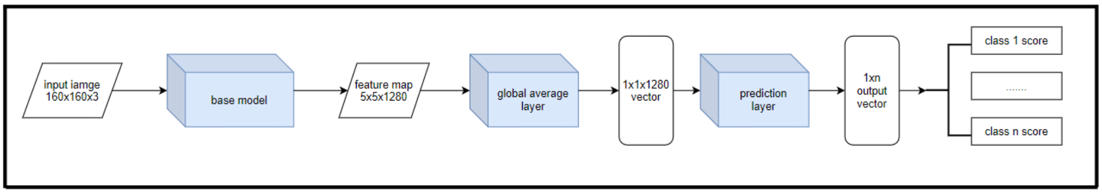

Figure 1-5 Network structure for this case

3.1.4 Base Model MobileNetV2

The transfer learning model in this solution is based on MobileNetV2 [7] pre-trained model. MobileNetV2 is developed by Google and it has been trained on more than 1.4M images with 1000 labelled classes and it is open-source.

Table 1.1-1 Parameters in SSD, YOLO, MobileNetV1, MobileNetV2

Network	Params

SSD300[8]	36.1M
SSD512[8]	36.1M
YOLOv2[9]	50.7M
MNet V1 + SSDLite[10]	5.1M
MNet V2 + SSDLite	4.1M

According to the datasets provided in this AM problem, the layers of the neural network model in this solution consist of some layers from the MobileNetV2 and additional layers for predictions that could fulfil the requirements in the manufacturing processes. MobileNetV2 can be downloaded from the internet by using the TensorFlow API tf.kera.applications. Once the base model is prepared, the next step of adapting the base model for this project is ready to perform.

3.1.5 Introduction to TensorFlow

TensorFlow is an open-source library developed and supported by Google which is widely used in building neural networks in the deep learning area. It uses graphs and tensors to optimize the computations and allows developers to use GPU to accelerate the training progress for large and complex models with massive parameters. There is a large and supportive community of TensorFlow so it is relatively easy for the group to build the transfer learning model. In addition, TensorFlow can provide the results in graphs and diagrams which make it easier to evaluate the performance of the model and to verify the satisfaction of the requirements of this project. In conclusion, TensorFlow with python will be used to train and evaluate the transfer learning model in this project.

3.2 Aims and Objectives

The aim of the project is to apply deep learning method through the images taken by processing cameras to identify the defects during the AM process. The CNN model is built by transfer learning method through a base model. The model will also be evaluated by different levels of criteria and verified to satisfy the requirements of the customers.
The objectives of this project are:

1. Understand and analysis technical requirements from customers in this case
2. Produce efficient project management for the group to finish this project on time
3. Do literature review on applying deep learning method for image classification on AM processing and using transfer learning method to implement a CNN model
4. Collect the data from customers and reformat the data set to be well prepared for software implementation

5. Develop a transfer learning model on input images and verify the performance on the test data set
6. Evaluate the performance of the model under different criteria to prove the model has met the requirements and it is capable to be applied on the AM process in the industrial field
7. Make conclusions on this project and propose future work to improve the model

3.3 Literature review

Additive manufacturing (AM) is a production process that uses materials to build products layer by layer. A very typical example is 3D Printing, AM works well on building products of complex shapes. In 2009, Bourell et al.[14] presented the AM roadmap at a workshop and explored the key areas of AM, such as design, process modelling and control, materials, processes and machines, biomedical applications, and energy and sustainability applications. In 2010, Frazier[15] published "Direct digital manufacturing (DMM) of metallic components: affordable, durable, and structurally efficient aircraft," the result of the workshop, the results on the surface of the specific technical challenges:

1. Innovative structural design

2. Qualification and certification

3. Maintenance and repair

4. Quantitative objectives in the field of science and technology DMM.

Among them, the highest level of research results includes: high priority should be given to in-process, monitoring, monitoring and control. Variability between machines must be understood and controlled. This means that the monitoring system is important to AM technology. And the importance of this is widely recognized.

However, in additive manufacturing, many factors have huge influences on the production results. Oliari et al[2]. proposed that in addition to the influence of laser power, walking speed and wire feeding speed, overlapping direction of layer-by-layer technology of the LMD process would also affect the production results. Through their research, the surface quality of the material, uniformity of the weld material, and the combination between the thin layers will be affected by these factors. According to these studies, additive manufacturing needs to be strictly monitored during the manufacturing process. Compared with the traditional manual monitoring, machine monitoring will be more efficient and low cost.

In order to implement the method of machine monitoring, all failures in the production process need to be understood. Dominic Gallello [3] mentions glitches, cracks, and high distortion that can be used as learning experiences. By analyzing the characteristics of these features, the performance of the machine learning model will be greatly improved. In order to realize the immediate detection and correction of defects, it is necessary to establish a failure detection method. One detection method was proposed by Nestor and Stephen [3], who used optical morphological and time-scale temperature measurements to detect defects. The second approach is deep learning using visual data. According to Gobert and Christian's research, they used a high-resolution digital SLR (DSLR) camera to obtain image data, used a deep learning method to extract features between layers, and then used binary classification technology [3] to evaluate these features. Because these defects are hard to see with the naked eye, Jack Francis et al.[6] used big data from laser-based additive manufacturing to training a new deep learning model. In order to accurately extract the characteristics of defects in the data, they used the camp-bp model, which can accurately predict distortion and improve the accuracy of distortion prediction. However, this approach requires the method of big data. In this project, this model cannot be used due to the limited amount of data. To solve this problem, a shape deviation modelling scheme was proposed by Longwe Cheng, Fugee Tsung, and Andi Wang et al. The scheme mentions: modelling the shape deviation as a transfer learning problem, the shape independent error, and the shape-specific error are well detected. With the help of statistical transfer learning, the problem of AM fault detection can be solved better.

In conclusion, there are basically two methods to solve this problem. The first is done by Nestor and Stephen G. L, they use the optics-based morphology and temperature measurements on time scales to detect the defects[3]. The other is deep learning, using the vision data. According to Gobert and Christian[3], they used a high-resolution digital single-lens reflex (DSLR) camera to get image data, using deep learning methods to extract features between layers then evaluated the features using binary classification techniques. In Y. Li, H. Yan, and Y. Zhang 's [6] work, they used deep learning methods to evaluate the material performance in laser AM. In Razaviarab's [5] work they used a closed-loop machine-learning algorithm to improve the process of AM. In Francis Jack's[6] work they use deep learning and big data to predict the distortion in LMD-W manufacturing. Since limited data were given to train our model. The transfer learning method will be used to build the model in this project.

4 Technical approach

This chapter introduces the technical approach implemented in this project for classifying the AM images in detail. The code of this project runs in Pycharm. This software is very friendly to python programming. It can quickly install the required library functions and toolkits through the built-in command terminal. The configured operating environment is TensorFlow built by Anaconda.

4.1 Data preparation

The first step to implement is to rearrange the image data set under the customer requirements. In this case, all data images are classified into three classes, which are ‘dripping’, ‘stub’, and ‘normal’.

For the efficiency of reading these data, all images which are going to be used are put in three subfolders, which are in the same sequence as the labels.

By using python's ‘pathlib’ function, read all images and label them as shown in table 2.2-1.

Table 2.2-1 data labels and related classes

label	class
0	dripping
1	stub
2	normal

Examples of raw train data with their labels are shown as in the figure 2-1 below.

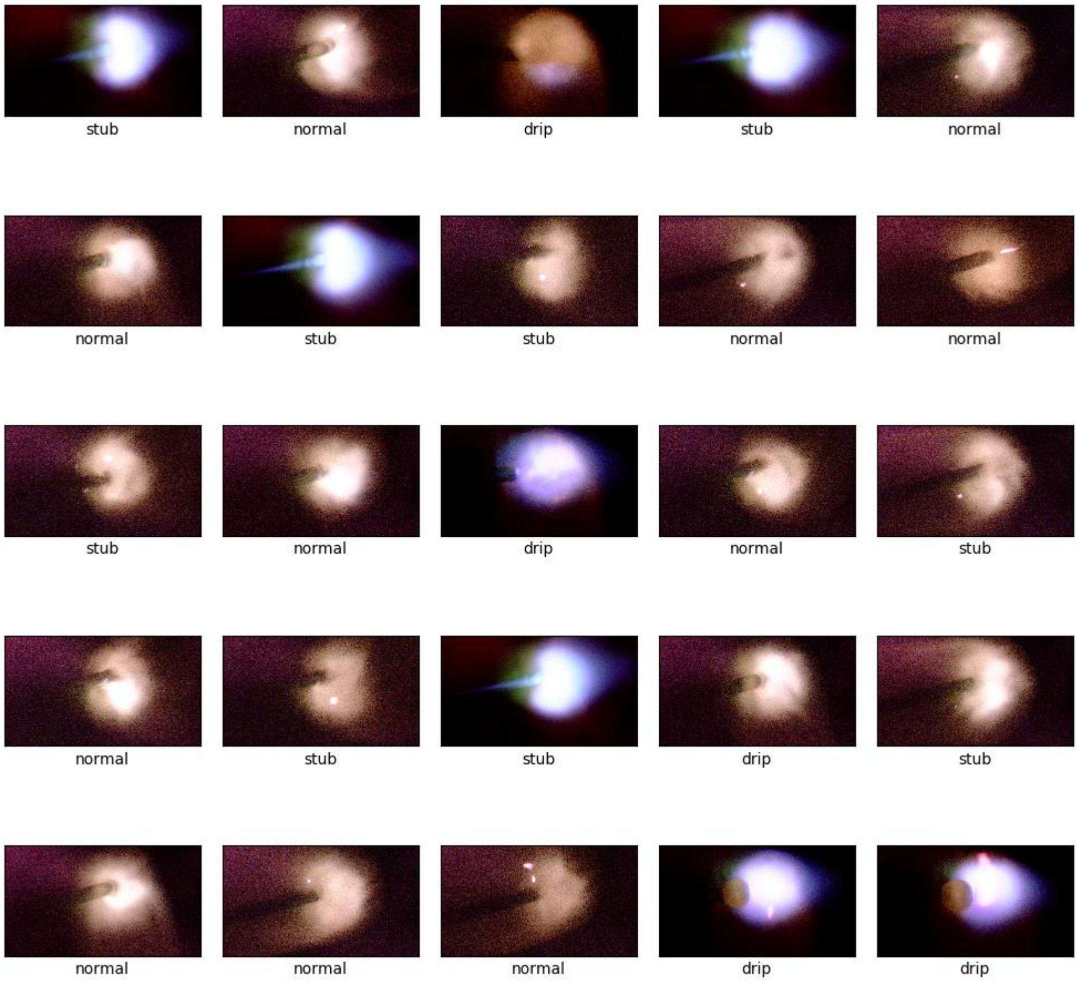

Figure 2-1 Examples of input image data for model training

The next step is to shuffle all the image data and split them into three different data set used for training, validation, and test. In this project, the team chose 80% of all as training data set, 10% data as validation data, and another 10% data saved for testing the model performance. TensorFlow 2.0 provides an efficient function to deal with the data with an API called 'tf.data'. Taking advantage of the function, all data used for training, validation, and testing can be reformatted together into a fixed input size and input channels to a range of [-1,1].

4.2 Model implementation

After preparing the data for training and testing the base model, several technical steps need to be carried out to perform further implementation.

1. In order to get the best performance in this model. Unfreeze the top layers from MobilenetV2 and set the parameters to be trainable.
2. To generate predictions from the block of features, average over the spatial 5x5 spatial locations, using a tf.keras.layers.GlobalAveragePooling2D layer to convert the features to a single 1280-element vector per image.
3. Apply an output layer to convert these features into a single prediction per image. Since this model is designed for three classification tasks, create a layer with three units.

4. Set the learning rate to 0.0001.
5. Apply tf.keras.optimizers.SparseCategoricalCrossentropy, which is a standard loss calculation for classification as the model loss function.

Then the new base model combined with the output layer is presented as the figure 2-2 shown below.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
mobilenetv2_1.00_160 (Model) (None, 5, 5, 1280)        2257984
_________________________________________________________________
global_average_pooling2d (Gl (None, 1280)              0
_________________________________________________________________
dense (Dense)                (None, 3)                 3843
=================================================================
Total params: 2,261,827
Trainable params: 2,227,715
Non-trainable params: 34,112
_________________________________________________________________
```

Figure 2-2 Base model architecture

Train this base model with the training data set and validation data set for 50 epochs. The results and their analysis will be shown in the next chapter.

To visualize the performance of this model, use plot function from python to present the learning curves of the training and validation accuracy/loss when using the MobileNet V2 base model as a fixed feature extractor.

The training accuracy and the validation accuracy will be shown in the plot as the epochs increasing. Training loss and validation loss will also be shown in another plot for the team members and customers to evaluate the performance of this model.

Analysis of the results and optimization methods along with the plots will be presented in the next chapter.

5 Simulation results and analysis

5.1 Final Result

Figure 3-1 shows the loss and accuracy achieved by the model during the 50 epochs training process using the LMD-w Manufacturing data set as the training set and validation set.

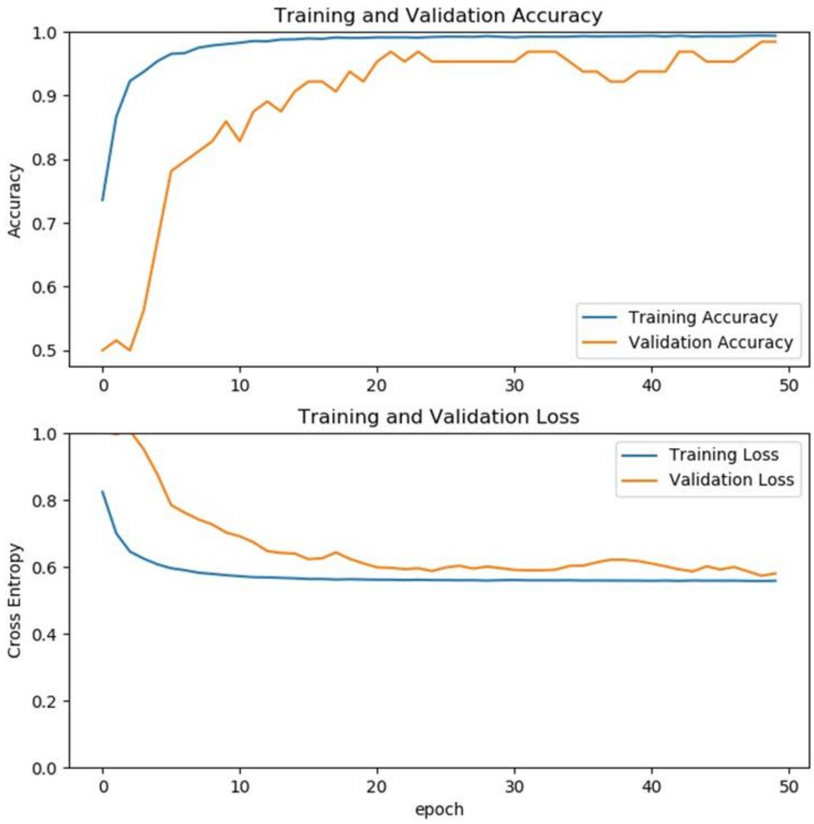

Figure 3-1 Accuracy and Loss of the Training and Validation

Each epoch means that all samples of the training set need to be input once on the model. In order to save training time as much as possible, each input is called step or batch and consists of 32 samples. The weight of the model would be updated at the end of each step. At the end of each epoch, verify the parameters learned by the model by calculating the value and accuracy of the loss function. The loss function uses the "Sparse Categorical Cross entropy" provided by TensorFlow for multi-classification problems. This loss function can better evaluate the error between the predicted value and the label value of the multi-classification problem. As shown in Figure 3-1, our model loss function values have tended to converge on both the training set and the validation set, meaning that what the model has learned has stabilised.

At the same time, the loss of the validation set is constantly approaching the loss of the training set, because the accuracy of the validation set of the model at this time is already very close to the accuracy of the training set. The accuracy of the training set is as high as 98.5% at 20 epochs, and the accuracy of the validation set has reached 92% at this time. As the number of iterations further increases, the weights of the model are continuously fitted to the real model, and the accuracy of the final model on the validation set is as high as 98.4%, which can reach the accuracy of manual judgment. The calculation method of accuracy is shown in formula (1), where Nc is the number of all correctly classified samples, and Na is the number of samples.

$$\%ACC = 100 * \frac{N_c}{N_a} \#(1)$$

Software evidence of this project achievement is shown in Figure 3-2. The code of this project runs in Pycharm. This software is very friendly to python programming. It can quickly install the required library functions and toolkits through the built-in command terminal. The configured operating environment is TensorFlow built by Anaconda. At the bottom of the software is the results of the model after completing 50 epochs training. The information in the fourth last line shows the final result of the model, and the accuracy rate is as high as 98.44% on the verification set. In addition, the test set is input to the trained model, and an accuracy rate of 98.09% is obtained, as shown in the fourth-to-last row in Figure 3-3. The samples in the test set did not participate in the training process. In other words, our model has learned about these errors that will occur in the process of identifying Laser Metal Deposition-wire, and this accuracy of the model has achieved high accuracy even on a data set that has not been trained at all.

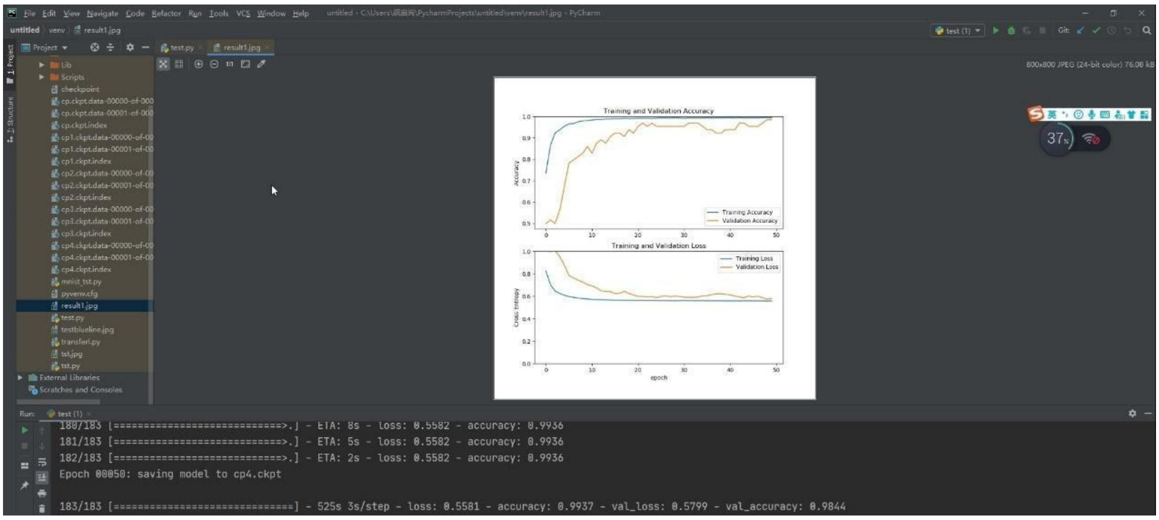

Figure 3-2 Project Achievement Software Evidence

In order to further verify that the accuracy of the model is robust, Figure 3-3 also shows the performance of the model in other model evaluation methods. These methods are recall, precision, and F1-score. Scikit-learn tool kit [3] was used to calculate all these three indicators. Considering that the model would solve the multi-class case, all model evaluation methods are based on the average of the score of each class. The precision is the ratio "tp / (tp + fp)" where "tp" is the number of true positives and "fp" the number of false positives. In a multi-classification problem, calculating the true positive of one of the classes is to calculate the number of classes that are correctly classified into this class. This class belongs to the positive class, while other classes are negative. The precision intuitively describes the ability of the classifier not to label as positive a sample that is negative. The best value is 1, and the worst value is 0. The recall is the ratio "tp / (tp + fn)" where "tp" is the number of true positives and "fn" the number of false negatives. The recall intuitively indicates the ability of the classifier to find all the positive samples. The best value is 1, and the worst value is 0. The F1 score can be interpreted as a weighted average of the precision and recall ("2 * (precision * recall) / (precision + recall)"), where an F1 score reaches its best value at 1 and worst score at 0. Our model achieves an average score greater than 0.98 in all three indicators averagely. In other words, the accuracy of the model is robust.

$$Precision = \frac{t_p}{(t_p + f_p)} \#(2)$$

$$Recall = \frac{t_p}{(t_p + f_n)} \#(3)$$

$$F1_{score} = \ 2 * \frac{precision * recall}{(precision + recall)} \#(4)$$

Figure 3-3 also shows the time required for each new input image to complete the prediction. Each figure completes the prediction that it takes an average of about 0.09 seconds for computer equipment equipped with a GTX1060 (6GB video memory) hard drive. This speed is sufficient for the model to work with the camera in the AM process in near real-time.

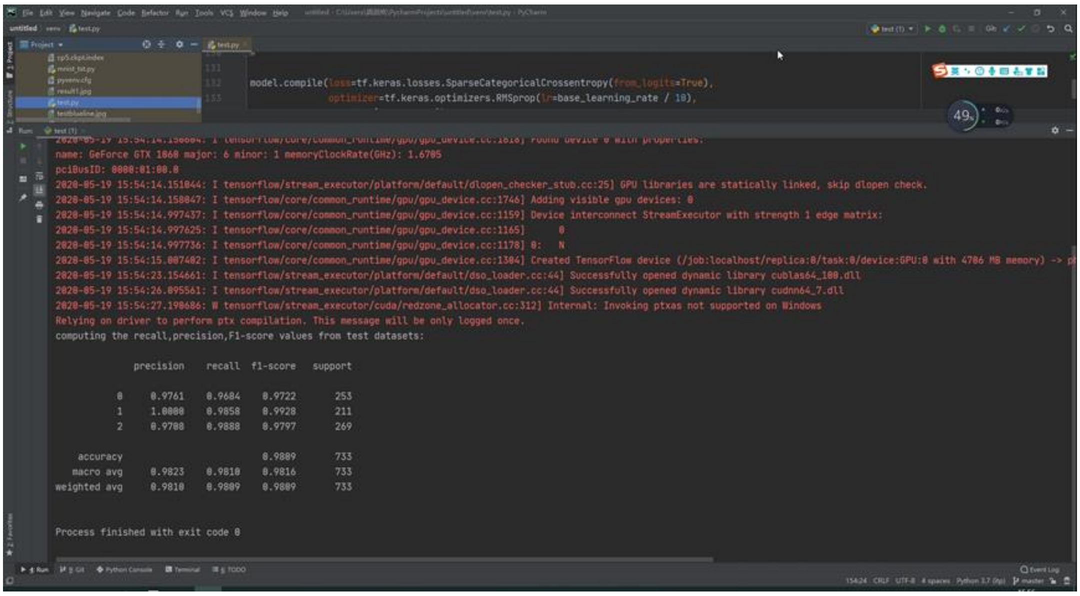

Figure 3-3 Model Evaluation Method-Precision, Recall and F1_Score

In order to more intuitively reflect the performance of the test samples on the model, Figure 3-4 shows 16 samples randomly drawn from the test set, where the title of each sub-picture is ground true label. The samples in the test set did not participate in the model training at all. These 16 pictures are input to the previously trained model, and the result of Figure 3-5 is obtained, in which the title of each sub-picture is estimated label. Since the model learned the features in the previous training with a very high accuracy rate, reaching 98.44%, it can be understood why only one of the 16 predicted values is wrong, as shown in the highlighted part of the green box in the sub-picture. By further observing and comparing the samples with the wrong prediction and other samples in both Figure 3-4 and Figure 3-5, the reason why this dripping picture is predicted to be a normal picture is that their texture characteristics and exposure degree are very similar, such as comparing with sub-picture 1. It is difficult for people to compare with the naked eye as well.

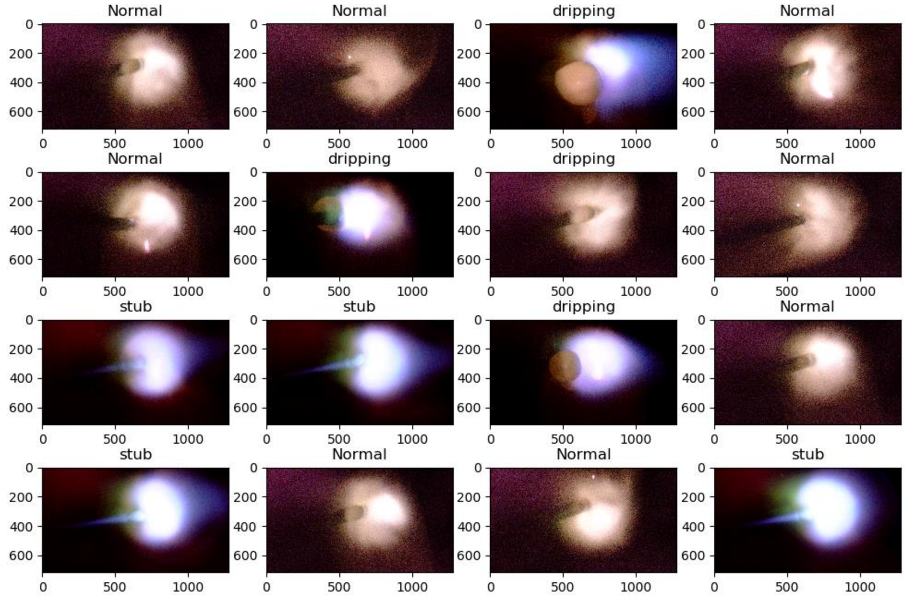

Figure 3-4 Ground True Images and Labels of 16 Samples

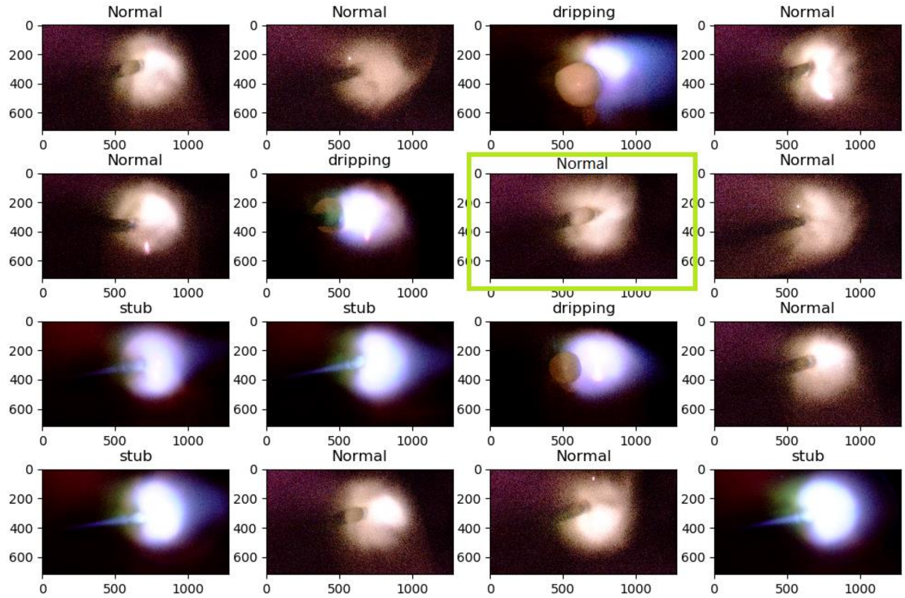

Figure 3-5 Predicted Labels of 16 Samples

5.2 Previous results and improvements

At first, the model was trained on the basis of the MobileNet V2 pre-trained model that comes with TensorFlow. The weights learned from the training model are from the ImageNet dataset

[11]. The ImageNet project is a large visualisation database for visual object recognition software research. Over 14 million image URLs were manually annotated by ImageNet. Since the data set is also an image type, theoretically, it is possible to start the training of the current data set based on the features of the parts learned in ImageNet through transfer learning. However, since the gap between the two data sets is very large, it can be found by comparing Figure 3-8 and Figure 3-4, so the result shown in Figure 3-6 is produced. At this time, the value of the loss function on the training set is continuously decreasing, but the value of the loss function on the verification set is not less than 1, in other words, overfitting has occurred. Therefore, Dropout was added to the convolutional layers in the model, a method of overfitting to inactivate a certain proportion of neurons. The results obtained are shown in Figure 3-7. Figure 3-7 is the same as the previous results of multiple epochs training, all curves show convergence, but the accuracy of the verification set is only 0.3, the loss function value is always higher than one. This means that the model has learned the effect of this dataset very poorly, and it is almost impossible to distinguish defects. This is because the weight obtained by pre-training is quite different from the weight obtained by the current data set. Therefore, the weights obtained by this pre-training were completely abandoned, and the parameters of the model only were learned from our data set. The results obtained are shown in Figure 3-9.

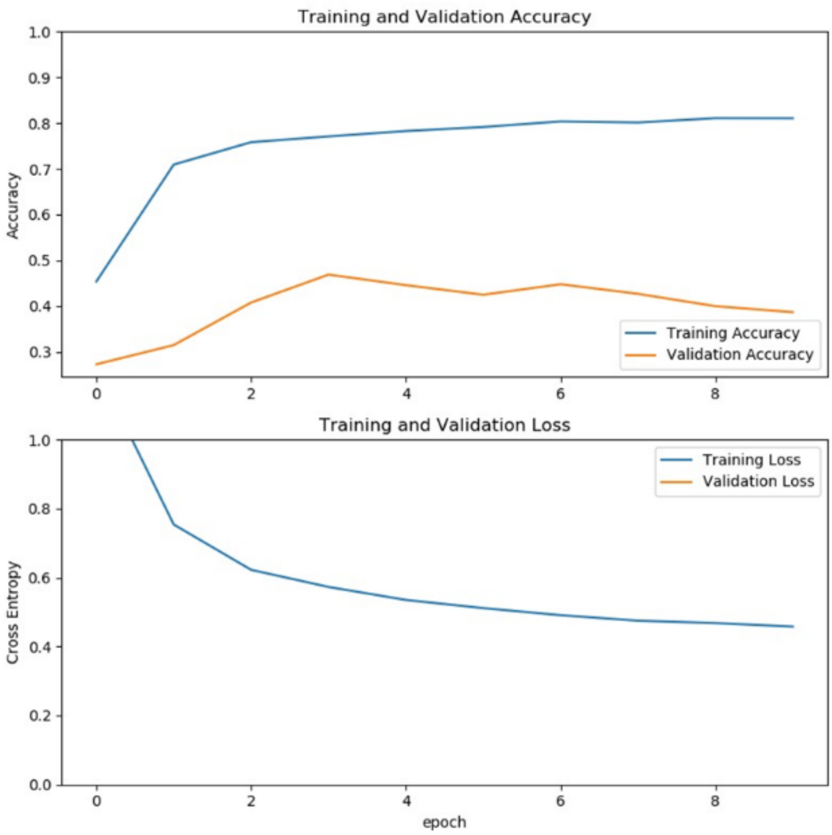

Figure 3-6 Model overfitting problem

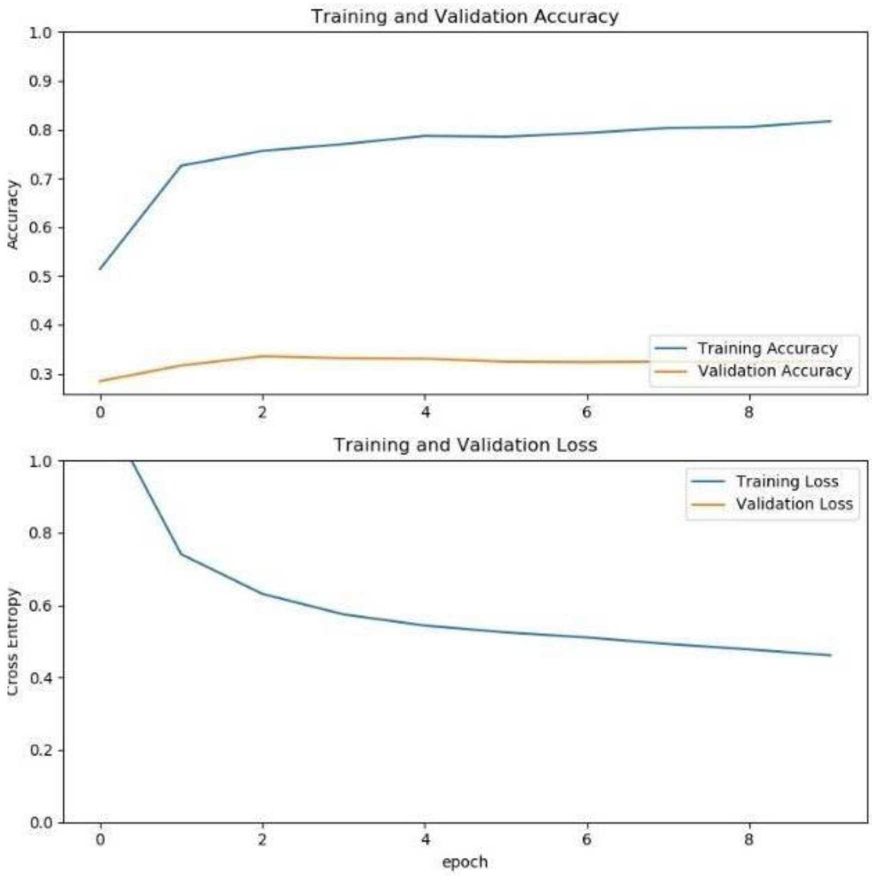

Figure 3-7 Utilising dropout to reduce the overfitting utilising dropout to reduce the overfitting

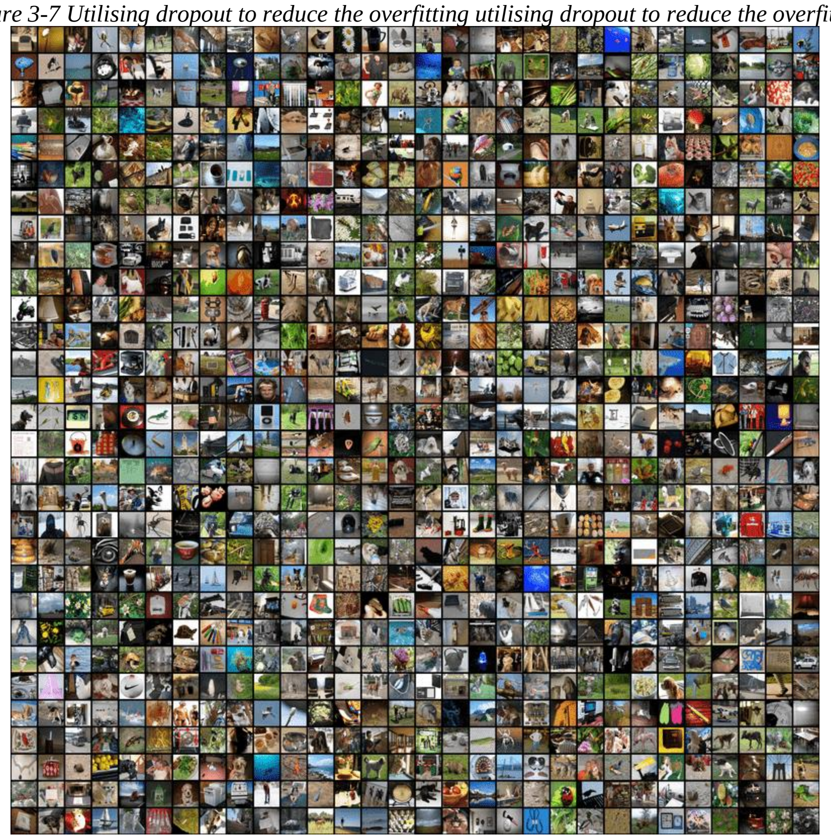

Figure 3-8 Samples from ImageNet dataset[12]

The results in Figure 3-9 also verify that when the data set corresponding to the pre-trained model is very different from the current data set, and transfer learning often results in the model not being able to learn the features of the current data set. The accuracy of the validation set reaches 0.82, which means that the model can learn certain features. At this point, the curves of loss functions have converged. However, the difference between the loss function value of the validation set and the training set is still about 0.2, which means the accuracy of the model can be further improved by increasing the training epochs. As shown in Figure 3-1, when epochs increase to 50, the accuracy of the model in the validation set reaches 0.98. In order to form a strict control, the samples in the training set, verification set, and test set in all experiments were not changed or reused.

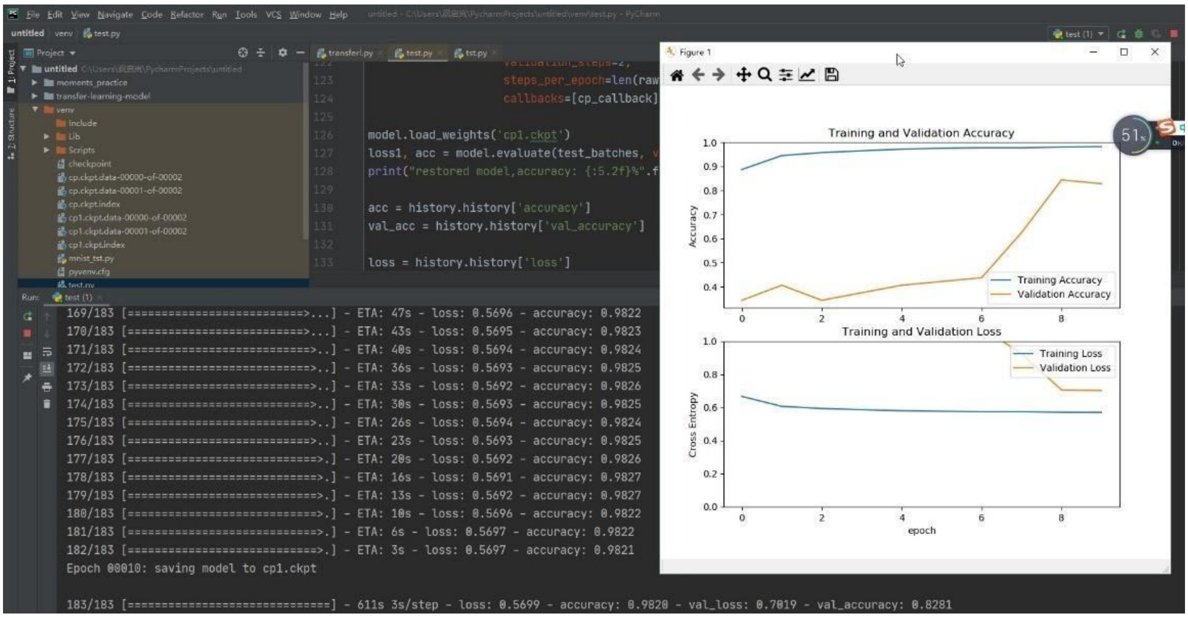

Figure 3-9 Loss and accuracy in epochs 0-10

6 Conclusions and directions for future work

Faulty detection by automatic methods instead of using human resources is essential for the efficiency and accuracy of AM processing. With a good performance deep learning model, machine learning methods are proved that it can be applied in such industrial cases.

6.1 Summary of methods and results

The model has satisfied the customer requirements proposed by the customers in most ways. With an accuracy of over 98.5% on the test data set, the transfer learning methodology applied to this project is capable of classifying the input images whether it is normal, dripping, or stubbing.

This model is developed using the TensorFlow in Python which meets the requirement of a software operating environment. Since this project is developed on the personal laptops of the group members, the project cost is 0 which also meets the requirement.

6.2 Suggestions for future works

6.2.1 Input data updating

In this project, the group picked all images into the three classes manually and then use them as the input data set. As a result, the model is only capable of classifying the input images in to three classes which are ‘normal’, ‘drip’ and ‘stub’. In real cases, there will be more kinds of images taken by the camera which this model will not be able to classify. For instance, sometimes the images are taken under different exposures. That can be a huge disturbing for the model to identify the images. In order to make the model fitting more in practice, this model can be trained with more different types with more accurate input images.

6.2.2 Base model updating

The group has chosen MobileNetV2 as the base model for transfer learning which is not a great model in this case. Choosing a more suitable base model for AM processing can significantly reduce the training cost and provide better performance with less trainable parameters.

7 Reference List

[1] D. Ding, Z. Pan, D. Cuiuri and H. Li, "Wire-feed additive manufacturing of metal components: technologies, developments and future interests", *The International Journal of Advanced Manufacturing Technology*, vol. 81, no. 1-4, pp. 465-481, 2015. Available: 10.1007/s00170-015-7077-3 [Accessed 8 March 2020].

[2] Additive Manufacturing of H11 with Wire-Based Laser Metal Deposition Oliari, Stella ; D'oliveira, Ana ; Schulz, Martin ISSN: 0104-9224 ; E-ISSN: 1980-6973 Soldagem e Inspecao, Oct-Dec 2017, Vol.22(4), p.466

[3] Application of supervised machine learning for defect detection during metallic powder bed fusion additive manufacturing using high resolution imaging Gobert, Christian ; Reutzel, Edward W ; Petrich, Jan ; Nassar, Abdalla R ; Phoha, Shashi **ISSN:** 2214-8604 ; **E-ISSN:** 2214-7810 ; **DOI:** 10.1016/j.addma.2018.04.005 Additive Manufacturing, May 2018, Vol.21, pp.517-528 References

[4] Y. Li, H. Yan and Y. Zhang, "A Deep Learning Method for Material Performance Recognition in Laser Additive Manufacturing," 2019 IEEE 17th International Conference on Industrial Informatics (INDIN), Helsinki, Finland, 2019, pp. 1735-1740.

[5] Razaviarab, Nariman, Safura Sharifi, and Yaser M. Banadaki. "Smart additive manufacturing empowered by a closed-loop machine learning algorithm." *Nano-, Bio-, Info-Tech Sensors and 3D Systems III*. Vol. 10969. International Society for Optics and Photonics, 2019.

[6] Francis, Jack, and Linkan Bian. "Deep learning for distortion prediction in laser-based additive manufacturing using big data." *Manufacturing Letters* 20 (2019): 10-14.

[7] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 4510–4520, 2018, doi: 10.1109/CVPR.2018.00474.

[8] W. Liu et al., "SSD: Single shot multibox detector," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9905 LNCS, pp. 21–37, 2016, doi: 10.1007/978-3-319-46448-0_2.

[9] C. World, "Yolo V2.0," Cvpr2017, no. April, pp. 187–213, 2017, doi: 10.1142/9789812771728_0012.

[10] A. G. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017.

[11] N. Yurtoğlu, " ImageNet," *History Studies International Journal of History*, vol. 10, no. 7, pp. 241–264, 2018.

[12] "Figure 2f from: Irimia R, Gottschling M (2016) Taxonomic revision of Rochefortia Sw. (Ehretiaceae, Boraginales). Biodiversity Data Journal 4: e7720. https://doi.org/10.3897/BDJ.4.e7720."

[13] S. Rigaud, "https://www.app.pan.pl/article/item/app20120056.html," *Acta* Palaeontologica Polonica, 2013.

[14]D.L. Bourell, M.C. Leu, and D.W. Rosen, Ed., Roadmap for Additive Manufacturing, University of Texas at Austin, Austin TX, 2009

[15]W.E. Frazier, "Digital Manufacturing of Metallic Components: Vision and Roadmap", Solid Free Form Fabrication Proceedings, University of Texas at Austin, Austin TX, 2010, p 717–732

[16] "Deep Learning," *Coursera*. [Online]. Available: https://www.coursera.org/specializations/deep-learning. [Accessed: 27-May-2020].