Research Progress in Mechanical and Manufacturing Engineering Vol. 3 No. 1 (2022) 884-892
© Universiti Tun Hussein Onn Malaysia Publisher's Office

RPMME

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/rpmme
e-ISSN : 2773-4765

Visual Inspection of Casting Product using Convolutional Neural Network

Mohamed Syahrir Azhar[1], Low Cheng Yee[1]*

[1]Faculty of Mechanical Engineering & Manufacturing,
Universiti Tun Hussein Onn Malaysia, Parit Raja, 86400, MALAYSIA

*Corresponding Author Designation

DOI: https://doi.org/10.30880/rpmme.2022.03.01.094
Received 01 Dec 2021; Accepted 01 April 2022; Available online 30 July 2022

Abstract: At the end of manufacturing of casting process, it is crucial to have a process which inspect if it has defects or not. Conventional method for inspection uses naked eyes which is prone to misclassify the product. The objectives for this study are to identify defects by using Convolutional Neural Network (CNN) simultaneously reduce the number of misclassification and to evaluate the performance of the CNN by comparing it with other image processing method. This approach is only applicable if the product is circle in shape and by using Python programming language for the codes. CNN is machine learning that specific in image processing. To build a fully functioning algorithm, the researcher has used google colab as the notebook and it should consist of 3 layers called convolutional layer where it extract all the feature contains in the raw image, pooling layer will reduce the size of the feature map to lessen the complexity for the next layers, and fully-connected layer is to combine all features to become an output. There are six criteria used when comparing CNN and ResNet. Final results prove that ResNet has a better performance as it is much simple to construct the algorithm because there is only 5 line of codes to import all libraries into the notebook while in CNN, the researcher need to start from scratch. The accuracy for determining the product are not much different compared to CNN. The accuracy for ResNet and CNN are 99.86% and 99.72%. For future researcher it is recommended to do a research related to Recurrent neural Network (RNN). RNN is commonly used in speech recognition.

Keywords: Convolutional Neural Network (CNN), Image Processing, Residual Network (Resnet).

**Corresponding author: cylow@uthm.edu.my*

1. Introduction

Metal casting can be described as a technique in which molten metal is poured into the mould cavity and let undergo drying process which turns it from liquid state into a solid-state [1]. In any manufacturing process, defects or flaws must be avoided as they will cause the whole batch of product being rejected. While doing metal casting process, defects which are likely to occur are misrun, cold shut, shrinkage cavity, and microporosity. To counter this problem, there come the quality control (Q.C) section.

Q.C section usually placed at the end of the manufacturing process before the product is handed to the customer. There are two method in Q.C section which non-destructive testing (NDT) and destructive testing. NDT can be divided into six categories which start with visual inspection, liquid penetrant testing, magnetic particle testing, electromagnetic testing, radiography, and ultrasonic testing [2]. This study will focus on NDT visual inspection which can be described as a form of evaluation by using human naked eyes [3]. But, due to humans' nature of doing mistakes, this method is prone to misclassify the product by stating defective product as an OK product. Not only that, this conventional method also required more worker to tally with the production.

To overcome the problem of misclassification and reduce the number of workers, a machine learning algorithm was constructed using a Python programming language. Essentially, machine learning is machines that follow the instruction given by humans using a specific instruction which is an algorithm that describes the task that needs to be done [4]. There are many types of machine learning which Convolutional Neural Network is part of it. CNNs are widely used in image recognition, object detection, and image segmentation or in another word it is specialized in image processing [5]. CNN is qualified for the extraction identification of data [6]. Not only that, it is also less complicated to construct as it need a few amount of frameworks to set up the model [5].

1.1. Objectives

i. To identify the defect in casting product using image processing, Convolutional Neural Network (CNN)
ii. To reduce misidentify of the product during Quality Control (QC) process
iii. To evaluate the performance of the algorithm by comparing it with another method

1.2. Scope of study

i. Using Python as programming language.
ii. Product shape is circle.

1.3. Significant of study

There has been rapid growth in the manufacturing process, especially in the casting process due to the increase in demand. All manufacturers are going to increase their production rate by adding more worker, machines, raw materials, and larger factories. At the same time, company also need to keep their quality standard in order to survive for long term. By using so-called "Artificial Intelligence" (AI) or to be specific Convolutional Neural Network (CNN), this will automatically eliminate the problem of misclassification, reducing the number of workers simultaneously reduce the monthly expenses.

2. Materials and Methods

2.1. Importing libraries

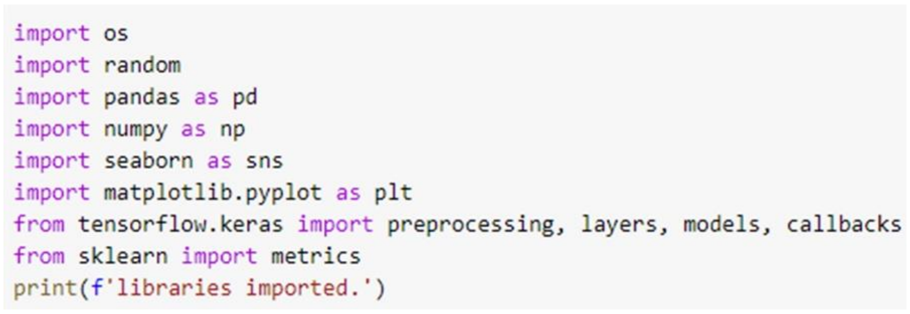

```python
import os
import random
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from tensorflow.keras import preprocessing, layers, models, callbacks
from sklearn import metrics
print(f'libraries imported.')
```

Figure 1: Libraries used in CNN algorithm.

Libraries used are operating system (OS), random, pandas, NumPy, seaborn, matplotlib.pyplot, TensorFlow.keras, and scikitlearn. The purpose of importing OS is to interact with the computer regardless it is Microsoft Windows, macOS, Linux, or Ubuntu. Next, a random library is used for generating a random number. Then, NumPy is used for working with arrays. For example, the model will "read" the image as an array of numbers which generated by Numpy library. Moreover, the Pandas library is used for data analysis and manipulation tools. After done with data analysing, the seaborn library is used for data visualization using charts, graphs, or maps. The last two lines tensorflow.keras and scikitlearnis are the libraries used in image processing neural network.

2.2. Specifying dataset's path

It is important to specify the path for the model to retrieve datasets easily. First thing is to upload the datasets into google drive. Next, write and execute these two lines in figure 2 below.

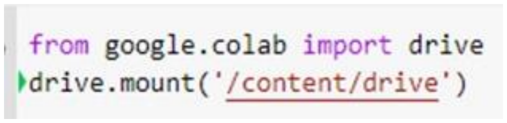

```python
from google.colab import drive
drive.mount('/content/drive')
```

Figure 2: Codes for mounting google drive with google colab.

After mounting google drive with the google colab, locations for each files need to be specify precisely. This is to make sure the model has no problem retrieving the datasets. Figure 3 below shows the path for all files. The main file that stores the datasets is given name "casting_data".

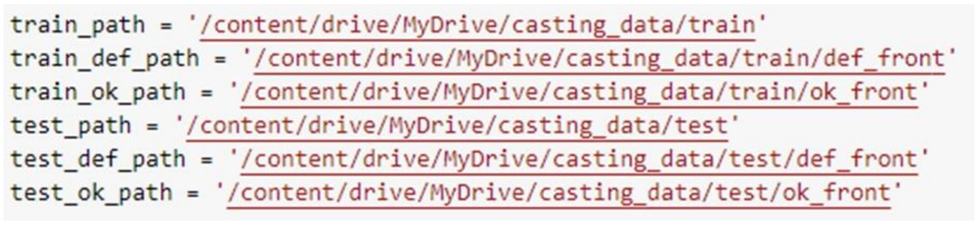
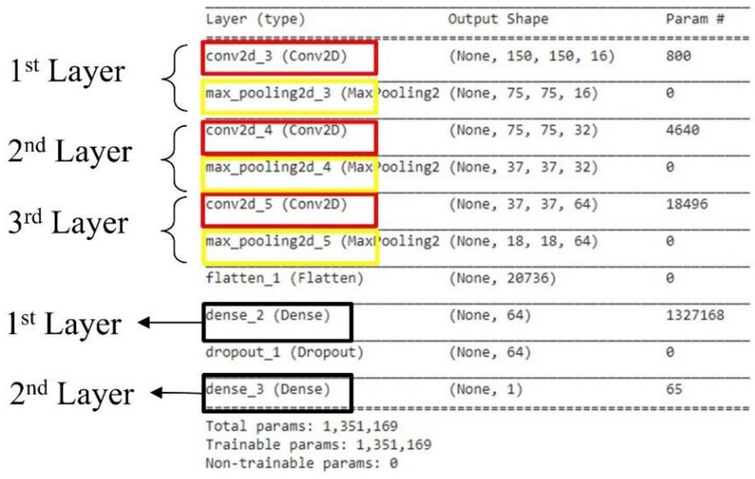

```python
train_path = '/content/drive/MyDrive/casting_data/train'
train_def_path = '/content/drive/MyDrive/casting_data/train/def_front'
train_ok_path = '/content/drive/MyDrive/casting_data/train/ok_front'
test_path = '/content/drive/MyDrive/casting_data/test'
test_def_path = '/content/drive/MyDrive/casting_data/test/def_front'
test_ok_path = '/content/drive/MyDrive/casting_data/test/ok_front'
```

Figure 3: Paths for all set of datasets.

2.3. Creating & compiling the model

This model will have a total of 5 layers. Three layers are used for convolutional (Conv2D) and max-pooling (MaxPooling2D). From there, the last output of max-pooling will undergo a flattening process and then be fed through two layers of the fully-connected or dense layer.



```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 150, 150, 16)      800
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 75, 75, 16)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 75, 75, 32)        4640
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 37, 37, 32)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 37, 37, 64)        18496
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 18, 18, 64)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 20736)             0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                1327168
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 65
=================================================================
Total params: 1,351,169
Trainable params: 1,351,169
Non-trainable params: 0
```

Figure 4: Layers to build a fully functioning CNN.

A distinct number of authors have observed that multiple layers of architecture sometimes give more accurate results [7]. An activation function is needed to help neurons to stimulate for the next layer by transforming it from linear to the non-linear regression model. It is then fed to the next layer as an input. In this model, an activation function is used for Conv2D and the first layer of fully-connected is rectified linear unit (ReLu). ReLu will perform a mathematical function that converts a negative input into zero. For the last layer of fully-connected, it is activated using the sigmoid function. The sigmoid function will transform the scores into the possibility of an image being classified.

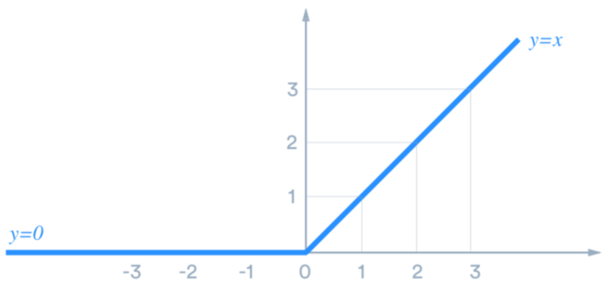
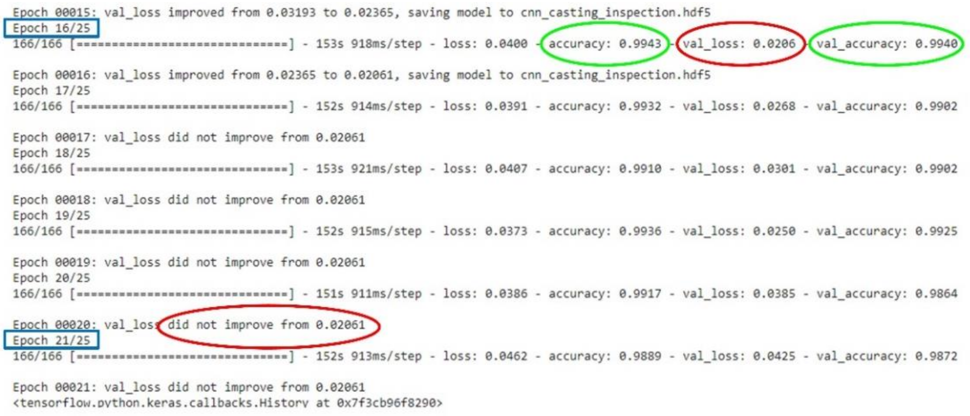


Figure 5: ReLu activation function that stimulate the neurons for next layer.

2.4. Training the model

In this model, the number of an epoch is set at 25. Epoch is the number of passes for the whole training dataset to complete. To determine the number of the epoch is by referring to the fitting. Model fitting is the measurement of how well the model classifies the validation dataset after it has been trained, the tolerance between the training dataset and validation dataset is marginal. If the model is under-fitting, this means that the number of an epoch is not enough and if it is over-fitting, the model has exceeded its practical number of an epoch. A good model will have perfect-fitting which means the model stops training just before the model starts to loss its gradient. But it is time-consuming to check it every time a number of the epoch is done. The best way to get this is by using *EarlyStopping* function. *EarlyStopping* means that the model will stop training the datasets when there is no improvement on validation loss (*val_loss*) after five consecutive epochs. To determine the number of steps for each epoch is by dividing the number of train datasets by a number of batches. Thus resulting in the number of steps per epoch to 166 steps. Next, *ModelCheckpoint* is used to save the model that gives the highest value of accuracy on training datasets and validation datasets. The model that has been saved will be used later for the test dataset.



```
Epoch 00015: val_loss improved from 0.03193 to 0.02365, saving model to cnn_casting_inspection.hdf5
Epoch 16/25
166/166 [==============================] - 153s 918ms/step - loss: 0.0400 - accuracy: 0.9943 - val_loss: 0.0206 - val_accuracy: 0.9940

Epoch 00016: val_loss improved from 0.02365 to 0.02061, saving model to cnn_casting_inspection.hdf5
Epoch 17/25
166/166 [==============================] - 152s 914ms/step - loss: 0.0391 - accuracy: 0.9932 - val_loss: 0.0268 - val_accuracy: 0.9902

Epoch 00017: val_loss did not improve from 0.02061
Epoch 18/25
166/166 [==============================] - 153s 921ms/step - loss: 0.0407 - accuracy: 0.9910 - val_loss: 0.0301 - val_accuracy: 0.9902

Epoch 00018: val_loss did not improve from 0.02061
Epoch 19/25
166/166 [==============================] - 152s 915ms/step - loss: 0.0373 - accuracy: 0.9936 - val_loss: 0.0250 - val_accuracy: 0.9925

Epoch 00019: val_loss did not improve from 0.02061
Epoch 20/25
166/166 [==============================] - 151s 911ms/step - loss: 0.0386 - accuracy: 0.9917 - val_loss: 0.0385 - val_accuracy: 0.9864

Epoch 00020: val_loss did not improve from 0.02061
Epoch 21/25
166/166 [==============================] - 152s 913ms/step - loss: 0.0462 - accuracy: 0.9889 - val_loss: 0.0425 - val_accuracy: 0.9872

Epoch 00021: val_loss did not improve from 0.02061
<tensorflow.python.keras.callbacks.History at 0x7f3cb96f8290>
```

Figure 6: Accuracy for epoch number 16.

From figure 6 above, epoch no.16 yields the best result. Scoring 99.43% accuracy on the training dataset and 99.40% on the validation dataset. *EarlyStopping* can be seen when the model terminated the training session after epoch 21 due to no improvement in validation loss (*val_loss*). The value of error kept on increasing from 2.06% until 4.03% since epoch 16.

3. Result & Discussion

For performance evaluation of CNN is done by comparing it with other method of image processing called residual network (ResNet). ResNet is an improvise version of CNN that has been introduce in the 2015 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) contest by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun [8]. The researcher creates a bypass connection called identity mapping that links the output of a residual block to its input that will decrease the vanishing gradient effect simultaneously makes training an algorithm much better [9]. Based on figure 7 below, X is an input to the residual block and the value of X should be equal to the output of the residual block. If the value of X is not equal to the output, the function residual (X) will alter so that the input value is parallel to the output value. When input X = output, then the function residual (X) will be zero and the identity mapping will duplicate the value of input X. To better understand the concept of the residual block, imagining the neural network is required to predict whether the product is a defect or OK based on the images that have been augmented. If the predicted is “defect” but the actual is “OK” then the identity mapping will alter it so that the predicted is equal to the actual.

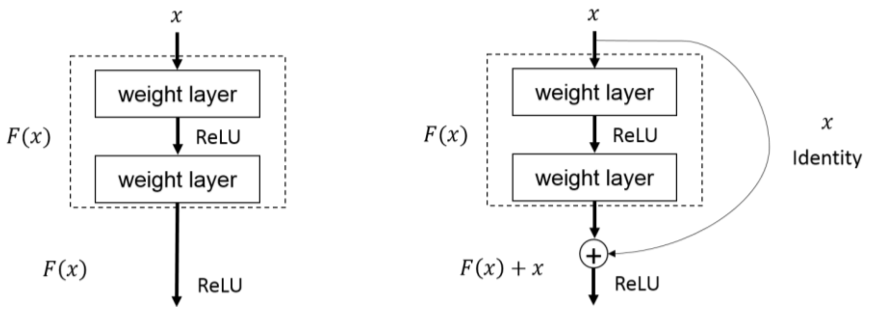


Figure 7: CNN block (left) compared to ResNet block (right).

Many types of Residual Network differ by their number of layers. For example, ResNet18, ResNet34, ResNet50, ResNet101, and ResNet152 which the number that comes after the name shows the number of parameter layers used for training the neural network [10]. In this paper, the researchers used ResNet34. In terms of architecture, it is basically the same as CNN, which consists of a convolutional layer, pooling layer, and fully-connected layer. The only thing that differentiates between CNN and ResNet34 architecture is the number of parameter layers which ResNet34 has 34 parameter layers while CNN only has 5 layers.

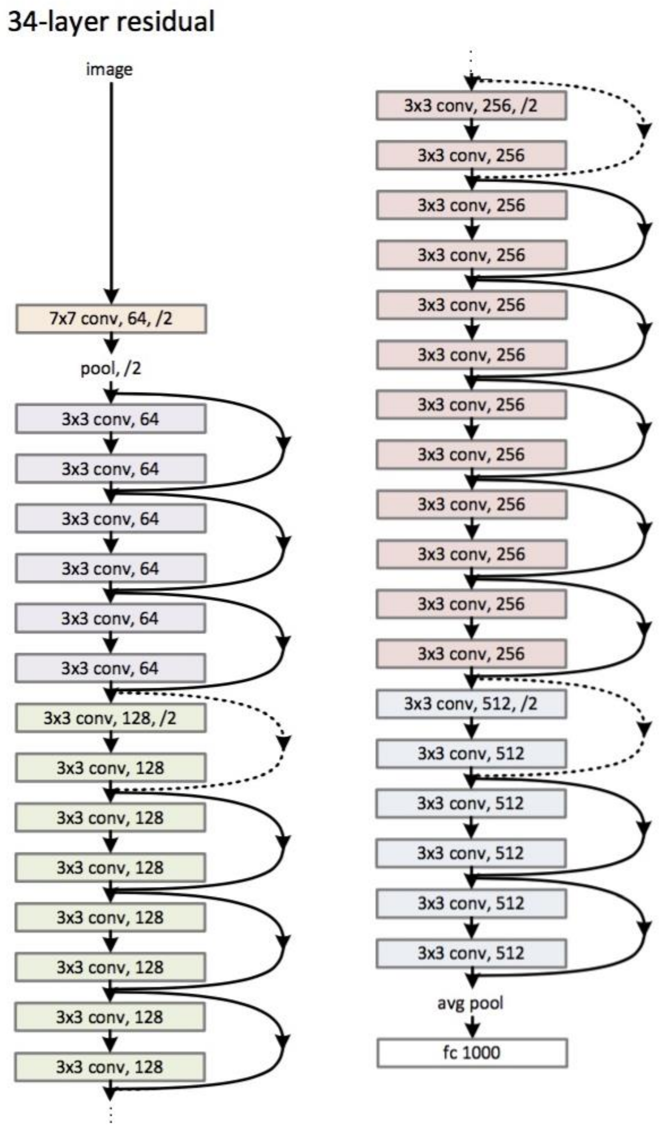
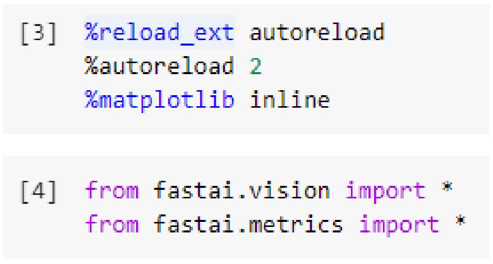


Figure 8: ResNet34 architecture that consist of 34 layers.

3.2 Discussions

In ResNet34, it uses IPython library which it is a better version of Python because Python is an open and general-purpose environment which means it is fairly basic to conduct a sophisticated task [11]. Advantages of using IPython can be seen at the early stage when importing libraries into the algorithm. By referring to figure 9 below, code used to import all necessary libraries are *%reload_ext* which serve the purpose of reloading the newest version of IPython, *%autoreload 2* configuration will reload all modules every time before executing the Python code, and *%matplotlib* inline will ensure that the matplotlib library to "communicate" bilateral with the coding which will render the figure in the notebook immediately after the coding has been executed. The symbol % before text indicates magic commands that are used for special features which the purpose of it is to enhance the IPython programming environment [12]. Next, importing image processing library to works with the ResNet34. Compared to CNN, it uses TensorFlow keras as the library in a neural network. In ResNet34, the researcher uses the Fast.ai V1 library. Fast.ai is a modern deep learning library that provides a high-level component that can rapidly and efficiently provide hi-tech results in a typical deep learning neural network [13].



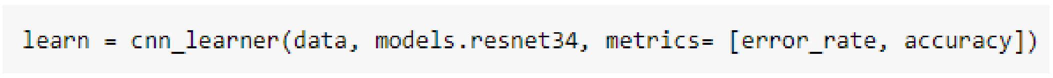
```
[3]  %reload_ext autoreload
     %autoreload 2
     %matplotlib inline

[4]  from fastai.vision import *
     from fastai.metrics import *
```

Figure 9: Modules and library used in constructing ResNet34

Next, in terms of retrieving the datasets, it is fairly same method used in CNN and The researcher will be using the same sets of datasets for both training and testing datasets from previous CNN experiment which consists of 6633 images for training and 722 images for testing which each of it consists of both defect and OK images.

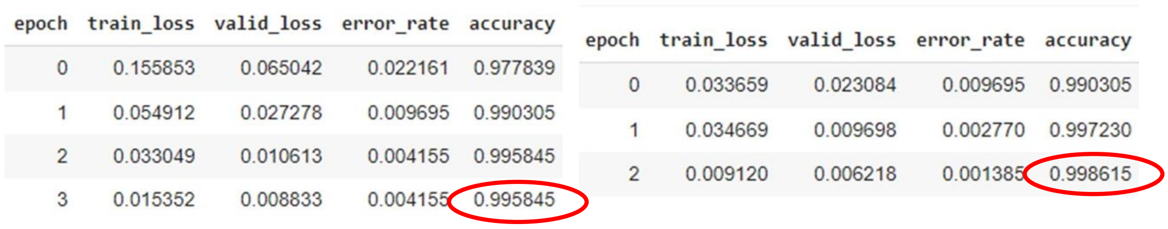
Now, the researcher has enter a phase of training the model. A "learner" is a specific name for the model that is being trained. In IPython, it has a pre-trained model. To "engage" the pre-trained model or ResNet34 architecture, it is a simple single-line code as shown in figure 10 below.



```
learn = cnn_learner(data, models.resnet34, metrics= [error_rate, accuracy])
```

Figure 10: Code to "engage" the pre-trained model

Once the architecture has been "engaged", the *fit_one_cycle* technique is used to train the ResNet34 architecture. This technique is chosen due to its incredible performance in terms of speed and accuracy. To better understand this, it is like choosing suitable exercises/drills for a specific subject because not all efforts are suitable for all subjects. For example, to enhance mathematical skills is to do more exercises rather than more reading the formula. The pre-trained model yields 99.58% accuracy. At this point, all layers except the last layer are read-only or called frozen. When the *fit_one_cycle* method is used on these frozen layers, the model is only trained for the unfrozen layer which is the very last layer that classify the images. The researcher will fine-tune the pre-trained model by unfreezing all these layers and simultaneously alter how the model classifies the images. Not only that but fine-tuning also covers the learning rate of the model. Based on figure 4.8 below, a new learning rate has been set using the *slice ()* function. After all, the modified pre-trained model undergoes training session again and yields 99.86%accuracy.



epoch	train_loss	valid_loss	error_rate	accuracy
0	0.155853	0.065042	0.022161	0.977839
1	0.054912	0.027278	0.009695	0.990305
2	0.033049	0.010613	0.004155	0.995845
3	0.015352	0.008833	0.004155	0.995845

epoch	train_loss	valid_loss	error_rate	accuracy
0	0.033659	0.023084	0.009695	0.990305
1	0.034669	0.009698	0.002770	0.997230
2	0.009120	0.006218	0.001385	0.998615

Figure 11: Accuracy for pre-trained model (left) and fine-tuned model (right).

Table 1: Comparison between CNN and ResNet34

	CNN	**ResNet 34**
Algorithm	Tedious	Easier
Image processing library	Keras	Fast.ai
No. of epochs	25 Epoch	7 Epochs (4 Epochs for pre-trained model & 3 epochs for fine-tuned model)
Run time	36 minutes	8 minutes
Accuracy	99.72%	99.86%
Misclassify	2 out of 722 images	1 out of 722 images

4. Conclusion

A company could not afford any misclassification of the end product as it will affect the company's revenue and reputation. Thus, the purpose of this study was to identify defects of casting products by using machine learning called Convolutional Neural Network (CNN) in order to reduce the number of misclassification products and also to evaluate the performance of the CNN. For the performance evaluation of CNN, a comparison between CNN and other image classification techniques called Residual Network (ResNet34) has been done.

The comparison between CNN and ResNet34 shows that ResNet34 gives the best performance between these two neural networks based on the six criteria that have been evaluated. It is proven that ResNet34 has a much simple way to create its algorithm as it has a pre-trained library which the only thing that needs to be done is to engage it by using only one-liner code. Whereas, in CNN the researcher needs to build the algorithm from scratch. In terms of accuracy, there is not much difference between ResNet34 and CNN which both accuracies yield 99.86% and 99.72%. All these advantages were due to ResNet34 being an updated version of CNN which was introduced in 2015. Based on the result achieved, it can be concluded that Residual Network (ResNet) has an advantage in image classification as it is straightforward to construct the architecture without compromising its results accuracy.

Acknowledgement

The authors would like to thank the Faculty of Mechanical and Manufacturing Engineering, Universiti Tun Hussein Onn Malaysia for its support.

References

[1] Kassie A A 2013 Minimization of Casting Defects *IOSR J. Eng.* **03** 31–8

[2] Idris J and Al-bakoosh A 2014 Application of Non-Destructive Testing Techniques for the Assessment of Casting of AA5083 Alloy *J. Adv. Res. Appl. Mech.* **3** 25–34

[3] Guide B 2015 Visual inspection Visual inspection 9–10

[4] Ayodele T O 2010 *New Advances in Machine Learning* ed Y Zhang (IntechOpen)

[5] Xu Y, Zhou Y, Sekula P and Ding L 2021 Machine learning in construction: From shallow to deep learning *Dev. Built Environ.* **6** 100045

[6] Nguyen T P, Choi S, Park S J, Park S H and Yoon J 2021 Inspecting Method for Defective Casting Products with Convolutional Neural Network (CNN) *Int. J. Precis. Eng. Manuf. - Green Technol.* **8** 583–94

[7] LeCun Y, Bottou L, Bengio Y and Haffner P 1998 Gradient-based learning applied to document recognition *Proc. IEEE* **86** 2278–323

[8] He K, Zhang X, Ren S and Sun J 2016 Deep residual learning for image recognition *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* **2016-Decem** 770–8

[9] Wu W H, Lee J C and Wang Y M 2020 A study of defect detection techniques for metallographic images *Sensors (Switzerland)* **20** 1–13

[10] Zhang Y, Wa S, Sun P and Wang Y 2021 Pear defect detection method based on ResNet and DCGAN *Inf.* **12**

[11] Pérez F and Granger B E 2007 IPython: A system for interactive scientific computing *Comput. Sci. Eng.* **9** 21–9

[12] Martins L F 2014 *IPython Notebook Essentials* (Birmingham: Packt Publisher)

[13] Howard J and Gugger S 2020 Fastai : A Layered API in Deep Learning 1