Vision system based on deep learning for product inspection in casting manufacturing: pump impeller images

S Taweelerd¹*, C C Chang¹ and G Y Tzou²

¹Graduate Institute of Mechatronic System Engineering, National University of Tainan, Tainan, 70005, Taiwan
²Department of Intelligent Automation Engineering, Chung Chou University of Science and Technology, Yuanlin Township, Changhua County 510, Taiwan

*Email: m10824011@stumail.nutn.edu.tw

Abstract. Products inspection is important issues in casting manufacture, because it is the final process before sending products to customers. To prevent a mistake from human operating error, vision systems are widely applied into this process nowadays. However, these systems still have some disadvantages which are sensitive to lightning and setup conditions. In this paper, the proposed approach for products inspection of submersible pump impeller images by the vision system based on deep learning with convolutional neural network architecture for casting manufacture is significant. It achieves the high accuracy of results as 99.7% on the top view of submersible pump impeller images dataset and requires less computational power and time. Moreover, it takes only 56.87 milliseconds for predicting one image. For proposing more details about this research, the submersible pump impeller images dataset is firstly presented. Subsequently, convolutional neural network, methods, evaluation and results are presented. Finally, all works in this study are summarized.

1. Introduction
Casting is one of manufacturing processes. In casting process, liquefied material, such as molten metal, is usually poured into a mold which contains a hollow cavity of the desired shape and allowed to solidify [1]. In the process, a casting defect is an undesired hollow cavity of the desired shape and allowed to solidify [1]. In the process, a casting defect is an undesired irregularity. There are many types of defect in casting such as blow holes, pinholes, burr, shrinkage defects, etc. Those defects are unwanted things in casting industry. For separation these defective products from non-defective, all industry has their quality inspection department. However, the main problem is the accuracy of inspection depend on human accuracy, this is not 100% accurate. A small mistake can be the rejection of the whole order. Thus, it creates a big loss to the company. For solving this problem, vision systems are the main choice to apply in this work. And also, deep learning with neural networks are growing up rapidly in the present. Thus, to improve efficiency of traditional vision systems and increase an accuracy in product inspection process, a vision system based on deep learning for product inspection is developed in this research.

1.1. Literature review
Industry 4.0 has opened the doors for deep learning to enter into the manufacturing arena with attempting to improve efficiency and quality check process. In many assembly lines vision systems are applied that can identify anomalies, read labels, count components and such like. However, current vision systems or traditional vision systems can fail in different setup conditions, this makes them a risk and so, quality control is not in any way aided or improved upon. For increasing performance of vision systems and solving these problems, deep learning is applied on the systems [2]. The main advantages of deep
learning with neural networks are more reliable, less prone to error vision system, tracking the components in real time regardless of lighting conditions and other constraints.

Among the growing up of computational power of graphical processing unit (GPUs) allows deep learning with neural networks to be flexibly expanded from depth and width [3, 4]. The GPUs are the main computational hardware that required in deep learning developments. It is much faster than central processing unit (CPUs) in complex mathematics or matrix calculation. Deep learning has begun to play a much more important role in smart manufacturing with vision systems and analytics. Furthermore, there are many available libraries, packages and improved neural networks such as the convolutional neural network (CNN) which is particularly efficient with images data [5]. Although we obtained many advantages from using deep learning in manufacturing floor, but one of the biggest issues is a dataset problem. Sometimes, the data can be difficult to collect.

The convolutional neural network has found popularity due to the fact that it does not require any manual feature extraction unlike traditional machine learning models [6]. It is also much better than traditional machine learning algorithms at high accuracy and less prediction time. However, there are some limitations such as it requires a large training data and high-performance computational requirements as mentioned above

2. Images dataset
This section introduces to images dataset, the dataset is the public dataset and free. It was downloaded from cloud data “Kaggle” [7]. By the images are top view of submersible pump impeller. It’s a product from casting manufacturing. This dataset contains total 7,348 images. These all are the size of (300 x 300) pixels grey-scaled images. In all images, augmentation already applied. The dataset was separated into two classes, training set and testing set. Each class has two subclasses, defective and non-defective. The preview of some image dataset is shown in Figure 1.

![Figure 1](image)

(a) (b) (c) (d)

**Figure 1.** Some of images dataset, (a) training defective, (b) training non-defective, (c) testing defective, (d) testing non-defective.

3. Convolutional neural network for inspection in submersible pump impeller images
The Convolutional Neural Network (CNN) is a neural network architecture specialized in image processing and pattern recognition. It is developed from the study of visual cortex structure in brain [8]. The CNN hierarchical structure allows multi-level image features to be extracted to achieve accurate pattern discovery [9]. Convolutional layer is most important block of a CNN structure. The convolutional layer is made of neurons that have learnable weights and biases (trainable parameter). Neurons in the first conv. layer are only connected to the pixels present in their receptive fields. In turn, each neuron in the second conv. layer is only connected to the neurons present in a specific local field in first layer [8]. This structure makes network to focus on low-level (local) features in first layer and subsequently these features are assembled into higher level features in following layer and then, this process continues. Therefore, the initial layers of CNN extract low-level (local) features of images like edges, curves, colors, etc. and the upper layers extract specific high-level features of images like shapes, etc. This structure is also common in natural real-world pictures that makes CNN networks to work well.
for image recognition. The mathematical equation for computation of neuron output in a convolutional layer is given below [8].

\[
Z_{i,j,k} = b_k + \sum_{u=1}^{f_h} \sum_{v=1}^{f_w} \sum_{k'=1}^{f_{n'}} x_{i',j',k'} \cdot w_{u,v,k',k}
\]

where, \(Z_{i,j,k}\) is the neuron output. This neuron is located in \(i^{th}\) row and \(j^{th}\) column in a feature map \(k\) of a convolutional layer \(l\). \(s_w\) is horizontal stride, \(s_h\) is vertical stride, \(f_w\) is width of receptive field, \(f_h\) is height of receptive field, and \(f_{n'}\) is number of feature maps present in preceding layer (layer \(l-1\)). \(x_{i',j',k'}\) is the output of neuron which is present in preceding layer \(l-1\). This neuron is present in \(i^{th}\) \((i')\) row and \(j^{th}\) \((j')\) column in feature map \(k'\) (or channel \(k'\) if it is input layer). \(b_k\) is bias for \(k\) feature map (in \(l\) layer). \(w_{u,v,k',k}\) is the connection weight between any of the neuron in map \(k\) of \(l\) layer and its input present at \(u^{th}\) row and \(v^{th}\) column, and the \(k'\) feature map.

Typical CNN architectures consist of few convolutional layers in start, then pooling layer, then few other convolution layers, then other pooling, and so on. At top of network, few fully connected layers and final decision layer is present. Unlike convolutional layers, neurons in fully connected layers do not have a limited receptive field. In addition to above layers, CNN networks usually consist of batch normalization and dropout layer [10].

4. Methodology
This section presents a classification model design, the model developed in this study is pure model based on submersible pump impeller images dataset and uses CNN architecture. It’s fully train from scratch. The data preprocessing is a second process in this section, this process images dataset is prepared for feeding to the created model. After two processes above, model training and validation are presented. In training process, the model is trained to learn to classify the submersible pump impeller images. At the same time model validation is working, accuracy and loss of the model are measured in this process. In the final, for finding an optimal model we perform optimization with series of architecture with different layers. As mentioned above, this study uses CNN architecture. Thus, we will only work on convolutional layers in this process. The flow chart of used approach is shown in Figure 2.

**Figure 2.** Flow chart of proposed approach.

4.1. CNN based model design
With CNN architecture, we started to design a first model by using one convolutional layer and two fully connected layers. After convolutional layer, max-pooling is used in pooling layer. ReLU is an activation function in convolutional layer and first fully connected layer. For last fully connected layer, sigmoid activation function is used. To give more details about these activation functions, we will present in sub-section 4.1.1. Hyper parameter of layers of the first model are given in Table 1.
Table 1. Hyper parameters of first model.

| Layer type      | Parameters                             |
|-----------------|----------------------------------------|
| Convolution     | 8 (3 x3) filters [ReLU]                |
| Max pooling     | 2 x 2 filter                           |
| Fully connected | 224 [ReLU]                             |
| Fully connected | 2 [Sigmoid]                            |
| Output          | 2                                      |

4.1.1. Activation function. An activation function is a function used in artificial neural networks which outputs a small value for small inputs, and a larger value if its inputs exceed a threshold. If the inputs are large enough, the activation function "fires", otherwise it does nothing. In other words, an activation function is like a gate that checks that an incoming value is greater than a critical number [11]. It is useful that adding the non-linearities into neural networks, allowing the neural networks to learn powerful operations. If the activation functions are removed from a feedforward neural network, the entire network could be re-factored to a simple linear operation or matrix transformation on its input, and it would no longer be capable of performing complex tasks such as image recognition.

- Rectified linear unit (ReLU) is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero. Advantages of using ReLU activation function are preventing the vanishing gradient problem, allows the network to converge very quickly, has no effect on size of volume. The mathematical of ReLU is given in equation (2).
- Sigmoid is an activation function that gives output between the range of 0 and 1. It is especially used for models where we have to predict the probability as an output. It also gives smooth gradient, prevents jumps in output values. These are main reasons of using it in last layer. The mathematical of Sigmoid is given in equation (3).

\[ f(x) = \max(0, x) \]  
\[ S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1} \]

where, \( x \) is the input to a neuron.

4.2. Data preprocessing

Originally the images dataset is stored in .jpg file type that unsuitable to feed into the model. Thus, these images must be converted to trainable variable type (tensor). For convenient working, there are lots of toolboxes that can be applied in this process such as Open CV, Matplotlib, PIL, etc. However, in this study we present Image data generator toolbox, this toolbox is the sub-toolbox of Keras [12]. It was made for data augmentation such as rotation, horizontal flip, vertical flip etc. The parameters of Image data generator toolbox in data preprocessing process are given in Table 2.

4.3. Model training and validation

Once we created a model and prepared data, model training and validation are ready to execute. Throughout these processes, we used Tensorflow and Keras framework [12, 13], it is an open-source software library for deep learning. The first step in training process is pre-setting a number of training epoch while optimizer, loss or cost function and metric selection are for validation process. This study we preset 20 epochs for training process, used adaptive moment estimation (Adam) [8] for optimizer that computes adaptive learning rates for each parameter during weight update. Furthermore, we used binary cross entropy loss function [14] to measure the inconsistency among two probability
distributions, mathematical is given in equation (4). And used accuracy [14] for evaluation, it is the ratio of correctly predicted observations to total events. Herein, true values are correctly predicted observations. An accuracy of binary classification can be calculated in terms of positives and negatives as equation (5).

During the model is trained, a very different way to regularize iterative learning algorithms such as learning rate, gradient descent is to stop training as soon as the validation error reaches a minimum. Thus, we applied Early Stopping, it is a callback function of frameworks that mentioned above. We can obtain more benefits of using this callback function such as prevent overfitting in training and testing data, reduce time consuming in training process.

After training process of the first model (one convolutional layer and two fully connected layers), we obtain an accuracy as 99.5% in training data, 97.8% in testing data while the loss is 3.1% in training data and 6.0% in testing data with using 10 epochs. Results of the model after training are shown in Figure 3.

### Table 2. Parameters in Image data generator toolbox.

| Parameter   | Value         |
|-------------|---------------|
| Target size | 300 × 300     |
| Color mode  | Grayscale     |
| Batch size  | 32            |
| Class mode  | Binary        |
| Shuffle     | True          |

\[
L = -\frac{1}{N} \sum_{i=1}^{N} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log(1 - \hat{y}_i) \tag{4}
\]

Where, \( \hat{y}_i \) is the \( i \)th scalar value in the model output, \( y_i \) is the corresponding target value, and \( N \) is the number of scalar values in the model output.

\[
Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}
\]

Where, \( TP \) is true positive values, \( TN \) is true negative values, \( FP \) is false positive values and \( FN \) is false negative values.

### 4.4. Model optimization

In this subsection, we perform experimentation with series of architecture with different layers to find an optimal model for product inspection in submersible pump impeller images. From the first model that created and trained, we obtained 99.5% of accuracy in training data and 97.8% in testing data with one convolutional layer and two fully connected layers. As we have seen, the accuracies are high but gap of accuracy between training and testing data is also high as 1.7%, that mean the model can work well in a seen data but not as it should be in a data that never seen.

This experimentation, we have increased the convolutional layer from one layer to two, three, four and five layers. In fully connected layers and activation function are fixed as the same with the first model. After that, these models are trained with the same dataset. From the use of Early stop-ping callback function during training process, the second model (two convolutional layers) and the fifth model (five convolutional layers) used 6 epochs while the third model (three convolutional layers) and the fourth model (four convolutional layers) used 10 epochs. Results of these models after training are shown in Figure 4.
After optimization, we have enough details to select the optimal model for our task. All of the models can give a high accuracy as over 97% in training data and testing data, while a highest loss of the model is 6.0% (first model, in testing data) and a lowest loss of model is 0.6% (fourth model, in testing data). Details about the first model, we have explained above. Thus, in this explanation, we will talk about details of the second model to the fifth model. From Figure 4(a), the highest accuracy is presented in testing data of the fourth, it is high as 100% while accuracy in training data is 99.6%. It tends to be overfitting in this model. The second highest accuracy is presented in training data of the third model, it is high as 99.7% and 99.6% in testing data that mean the gap between training data and testing data of this model is only 0.1%. Thus, in this study we selected this model (three convolutional layers and two fully connected layers) for our task. For others model left (second model and fifth model), the accuracies of them in both data are lower than the selected model and losses of them are also higher. Hyper parameters of selected model are given in Table 3.

| Table 3. Hyper parameters of selected model. |
|---------------------------------------------|
| Layer type       | Parameter setting       |
| Convolution      | 8 (3 x 3) filters [ReLU] |
Max pooling 2 × 2 filter
Convolution 16 (3 × 3) filters [ReLU]
Max pooling 2 × 2 filter
Convolution 16 (3 × 3) filters [ReLU]
Max pooling 2 × 2 filter
Fully connected 224 [ReLU]
Fully connected 2 [Sigmoid]
Output 2

5. Evaluation and results
This section has presented model evaluation with testing images data. As mentioned in section 2, Each data has two subclasses (defective and non-defective). For testing data, defective class has 453 images while non-defective class has 262 images. All of these images are fed into the selected model to predict classes of the images, by the model has never seen these images before. The model takes 56.87 milliseconds for predicting one image, and does not require any pre-processing of acquired images to be tested.

After evaluating, we obtained result as 99.6% of accuracy in prediction in testing data. Where the model can predict correctly 451 from 453 images in defective class and 261 from 262 images in non-defective class. The confusion matrix obtained at this point is shown in Figure 5. The few correct and wrong predictions made by the developed model are shown in Figure 6. The first row shows correct predictions by the model while the second row shows wrong predictions, where NOK represents defective class and OK represents non-defective class. True and predicted labels for each of submersible pump impeller image are present at the top of the images.

![Confusion Matrix](image)

Figure 5. Confusion matrix for the proposed model in prediction in testing data.
Figure 6. Images of evaluation of predictions made by proposed model: first row shows correct predictions in submersible pump impeller images, and second shows wrong predictions submersible pump impeller images.

6. Conclusion
In this study, we presented an approach for products inspection of submersible pump impeller images by using the vision system based on deep learning with convolutional neural network (CNN) architecture. The proposed approach achieved state of the art results on publicly available images data-set. It achieved an average accuracy of 99.7% by using only 10 epochs and 1.51 minutes for model training process. While for prediction, it takes 56.87 milliseconds for predicting one image. For proposing the model for products inspection, we perform experimentation with different layers of convolutional layer. With the raw image dataset, we proposed the convenient toolbox for data preprocessing process. This study can help for products inspection field and industry. The experiments showed that CNN architecture can achieve high accuracy. In future work, the defective products can be easily classified out from the non-defective and reduce human error in this working field.

Acknowledgements
Authors would like to acknowledge Mr. Ravirajsinh Dabhi and his researcher team from India for providing public submersible pump impeller images dataset. Their dataset is extremely useful and helps this research to be completed.
References

[1] Wales J D and Sanger L M, "Wikipedia," Wikimedia Foundation, 15 January 2001. [Online]. Available: https://en.wikipedia.org/wiki/Casting

[2] Kelly O and Jacqueline H 2019 Object Detection using Convolutional Neural Networks for Smart Manufacturing Vision Systems in the Medical Devices Sector 29th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2019) (Limerick)

[3] He K, Zhang X, Ren S and Sun J 2016 Deep Residual Learning for Image Recognition 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Las Vegas)

[4] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V and Rabinovich A 2015 Going deeper with convolutions 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Boston)

[5] Wang J, Ma Y, Zhang L, Gao R X and Wu D 2018 Deep Learning for smart manufacturing: Methods and applications Journal of Manufacturing 48 144-156

[6] Pathak A R, Pandey M and Rautaray S 2018 Application of Deep Learning for Object Detection Procedia Computer Science 132 1706-1717

[7] "Kaggle," Google LLC, 2010. [Online]. Available: https://www.kaggle.com/. [Accessed June 2020]

[8] Akram W M, Li G, Jin Y, Chen X, Zhu C, Zhao X, Khaliq A, Faheem M and Ahmad A 2019 CNN based automatic detection of photovoltaic cell defects in electroluminescence images Energy 189

[9] Alessandra C, Jianjing Z, Vittorio A, Fabrizia C, Robert G and Roberto T 2019 Machine learning-based image processing for on-line defect recognition in additive manufacturing CIRP Annals - Manufacturing Technology 68 451-454

[10] Sergey I and Cristian S 2015 Batch normalization: accelerating deep network training by reducing internal covariate shift Proceedings of the 32nd International Conference on Machine Learning (PMLR) vol 37 p 448

[11] LaFirenza M, "Deepai," 2017. [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/activation-function

[12] François C, "Keras.io," Google, 27 March 2015. [Online]. Available: https://keras.io.

[13] Google, "Tensorflow," Google, 9 November 2015. [Online]

[14] Goodfellow I, Bengio Y and Courville A 2016 Deep Learning MIT Press