Mask Defect Detection: A Journey through Real-World Applications of CNN-based Models*

Mask Defect Detection

A Journey through Real-World Applications of CNN-based Models

Cheng-Yuan Ho [1]

Department of Information Management, National Taiwan University, tommyho@ntu.edu.tw

Yong-Sheng Yang [2]

EXPETECH CO., LTD., ams7758@gmail.com

In 2019, the Covid-19 pandemic swept across the world, prompting people to wear masks as a necessary precaution when going outside. However, at the start of the outbreak, panic buying caused a severe shortage of masks. As a result, many factories expanded their production lines or started producing masks to meet the increased demand. However, various factors during mask production can lead to various defects, necessitating manual visual inspection to check for defects. Manual inspection is prone to fatigue and is also costly in terms of labor. Therefore, this study proposes replacing manual inspection with artificial intelligence to save a significant amount of labor costs. Using artificial intelligence for defect detection on the production line can save approximately 1-2 laborers, and a single computer can simultaneously detect defects in hundreds of images, reducing labor costs in factories.

This study investigates the use of five common convolutional neural network models for mask defect detection, using a total of 11,960 mask images for training, validation, and testing. Of these, 1,198 randomly selected images were used for final accuracy testing. The experimental results show that the best-performing model was InceptionResNetV2, with an accuracy of 99.58%.

CCS CONCEPTS •Computing methodologies ~Artificial intelligence ~Computer vision ~Computer vision tasks ~Visual content-based indexing and retrieval •Computing methodologies ~Artificial intelligence ~Computer vision ~Computer vision tasks ~Activity recognition and understanding •Computing methodologies ~Artificial intelligence ~Computer vision ~Computer vision tasks ~Scene anomaly detection

Additional Keywords and Phrases: Artificial Intelligence, Convolutional Neural Networks, Defect Detection, Mask, Smart Manufacturing

---

* Research supported by the National Science and Technology Council (NSTC), Taiwan, under Grant Nos. MOST 109-2221-E-468-008-MY3 and MOST 111-2221-E-002-197.

[1] Cheng-Yuan Ho was affiliated with Asia University, Taichung, 413, Taiwan, and is now with the Dept. of Information Management, National Taiwan University, Taipei, 10617, Taiwan (corresponding author; phone: +886-2-3366-1923; fax: +886-2-3366-1199; e-mail: tommyho@ntu.edu.tw).

[2] Yong-Sheng Yang was Cheng-Yuan Ho's master student at Asia University, Taichung, 413, Taiwan, and is now with Expetech Co., Ltd., Taichung, 408, Taiwan (e-mail: ams7758@gmail.com).

1 INTRODUCTION

In 2019, the outbreak of COVID-19 had a significant impact on almost the entire world. However, it also spurred the development of many artificial intelligence (AI) services that reduced the need for human labor and, consequently, the risk of infection and transmission. These AI services and applications are now commonly seen in everyday life, such as temperature monitoring at entrances, unmanned vehicles, and unmanned factories. While the first two applications are ubiquitous, the use of unmanned factories varies depending on each factory's product and needs. Additionally, due to the pandemic, masks have become a necessary item when leaving home. However, quality control during mask production still relies heavily on human labor, which can lead to fatigue and errors over time, resulting in a decrease in production efficiency and an increase in defective masks. Even with disinfection and protective measures, human negligence can still lead to contaminated masks. Therefore, the transformation of production lines into unmanned ones requires the use of AI for upgrading. This study attempts to use deep learning to detect mask defects in factories, hoping to use AI to recognize defective masks on the production line and reduce the use of quality control personnel.

To obtain image data of defective masks, this study collaborated with Yufa Technology Co., Ltd. [2] to provide defective masks and related equipment to the research team. However, the dataset was classified and filmed based on the experience and opinions provided by the Yufa Technology Co., Ltd., and will be described in detail in the research method section.

In recent years, with the development of AI [3-6], deep learning has become increasingly popularly applied [7]. The use of Convolutional Neural Networks (CNN) to solve image classification problems is particularly noteworthy [8]. This study aims to utilize the characteristic of deep learning, which can learn features automatically, to achieve a similar effect to automatic optical inspection (AOI) for defect detection [9]. Therefore, this study selects five models that have performed well on ImageNet in recent years [8, 10] as the defect detection models, attempting to use these models to achieve good performance in the defective mask dataset of this study, and to be applied in actual factories in the future.

2 RESEARCH METHOD

The research process of this study consists of three parts, as shown in Figure 1. Firstly, data collection was conducted, where masks were collected and classified, and initial data was obtained by taking pictures of each mask. Next, data preprocessing was performed to process the collected data to make it better suited for training in a neural network. Additionally, data augmentation was applied to enhance the accuracy of the model. Subsequently, the data was fed into the model for training, and the effectiveness of various models was compared using test data. Finally, the best-performing model was selected as the defect detection system.

2.1 Research Equipment and Environment

This study employed a desktop computer as the experimental equipment, and its hardware and environmental configurations are shown in Table 1. The experiment was conducted in the Ubuntu 20.04 environment using Jupyter Notebook for Python programming, in combination with tensorflow-gpu to use the GPU for model training to improve training and prediction speed. As the Ubuntu system itself does not include software such as Tensorflow-gpu, cuda, and cudnn, these software packages must be installed separately. Before installing tensorflow-gpu, Nvidia cuda and cudnn need to be installed first, and their versions must be matched with the version of tensorflow-gpu to be used. For more information, please refer to the official websites of Tensorflow and Nvidia.

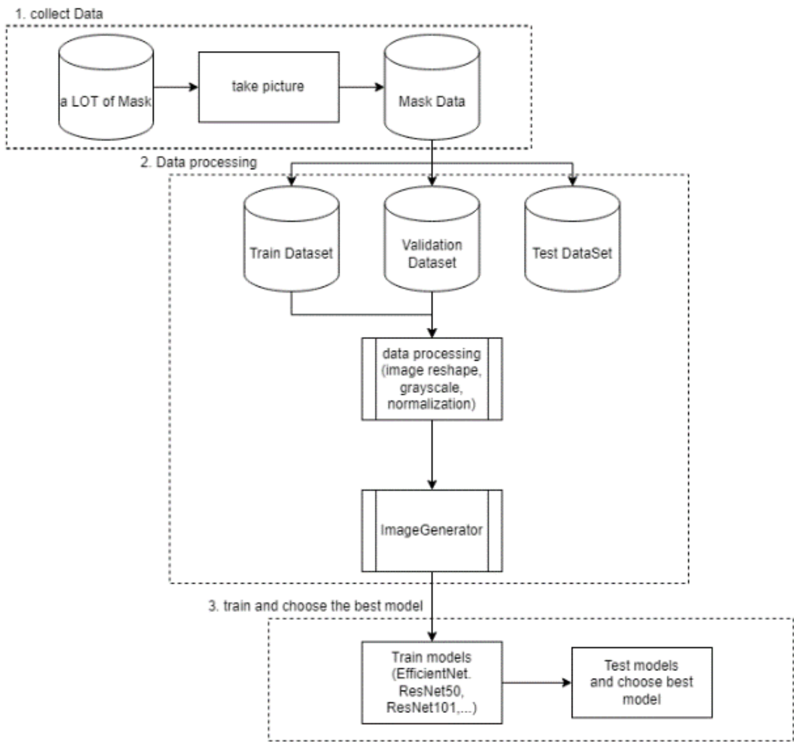

Figure 1: Research Flowchart.

Table 1: Computer Environment Configuration

Hardware Configuration	
CPU Processor	Intel i5-9400F
RAM Memory	32GB
GPU Graphics Card	Nvidia RTX3060 *2
System/Software Configuration	
Operating System	Ubuntu 20.04
Tensorflow-gpu	2.4.0
CUDA	11.0
cudnn	8.0

2.2 Data Collection

To obtain a large number of defective masks and normal masks, this study obtained various types of defective masks from Yufa Technology Co., Ltd. The normal and defective masks produced by the company were classified and made into a defect detection dataset. The masks were divided into four categories: normal, wire defect, ear strap breakage, and defective products as shown in Figures 2-5. Each category of masks was separated and photographed separately, and the classification criteria are shown in Table 2. In total, 11,960 images were collected in the defect detection dataset, which were then split into three datasets for training, validation, and testing. The split ratio used was 7:2:1 for each category.

Figure 2: Normal Mask.

Figure 3: Mask with Wire Defect.

Figure 4: Mask with Ear Strap Breakage.

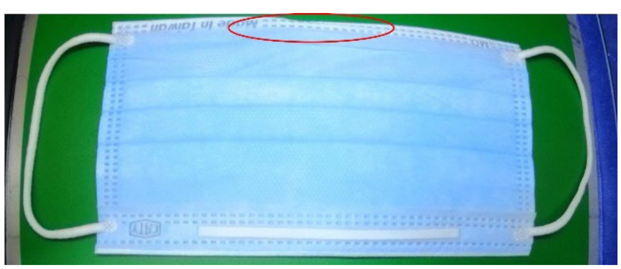

Figure 5: Defective Mask.

Table 2: Mask Defect Classification Criteria

Category	Mask Defect Classification Criteria
Normal	A mask without defects that can be used normally
Wire Defect	Wire displacement, misalignment, breakage, excessive length or lack of wire
Ear Strap Breakage	Ear strap breakage or lack of ear strap
Defective	Mask appearance defects, uneven fabric, but does not affect mask function

2.3 Data Preprocessing

Before training the model for defect detection system, in order to effectively teach the model correct features, the images need to be processed before training. In addition, in order to train the model better, this paper used the method of TensorFlow's ImageDataGenerator to augment the data, increasing the number of images from 11,960 to over 400,000, in order to help the model to be better trained.

The first step in processing the data is to resize the images, because the size of the images in the dataset used in this paper is 1920 x 1080, it is necessary to resize the images to enable faster training. In this study, the images were resized to 740 x 370 through the built-in function of ImageDataGenerator, and the masks were also processed into grayscale. This is because some parts of the masks in the data used in this study did not have rich colors, which might cause the model to learn color as a feature. Additionally, this paper also used the rescale method to convert the pixel values of the masks from 0-255 to floating-point numbers between 0-1, in order to prevent neuron death due to the pixel values being too large during training. The aforementioned methods were all completed within ImageDataGenerator, and horizontal_flip, width_shift_range, height_shift_range were also used to flip the images horizontally and make slight movements in the left, right, up, and down directions to prevent the model from learning fixed position features.

Finally, the training data was shuffled by the random function in ImageDataGenerator, and the batch_size parameter and other parameters were set to begin training and adjust the training parameters.

2.4 Defect Detection Systems

In this study, we selected five models, including ResNet101, ResNet152, InceptionResNetV2, InceptionV3, and EfficientNet, as the models for training the defect detection systems. These models are commonly used and readily available in TensorFlow 2.4. Additionally, the EfficientNet includes eight different models with varying network depths, widths, and resolutions. Among these eight models, we selected the B3 model to represent the EfficientNet and compare it with other models because B3 can run smoothly on the existing equipment and performs the best.

To facilitate the training process, we also adjusted the training parameters. We adjusted the input_layer to a size of 740 x 370 when inputting images into the model to make the images larger than those in a general dataset. This allowed the model to detect subtle differences in the masks. However, this size also reduces the training speed and increases the model's complexity, making it unusable for some models due to insufficient GPU memory.

Before training, we set three parameters: earlystop, reduce_lr, and optimizers. Earlystop is used to monitor whether the model is training successfully. If there are issues such as failure to converge, overfitting, or slow convergence, the training will be terminated. In this study, we set the learning rate to stop training if there is no improvement over 10 iterations by more than 0.0001. Reduce_lr is used to monitor whether the specified indicators have improved during the learning process. If there is no improvement, the learning rate will be reduced to improve the learning process. Additionally, earlystop and reduce_lr monitor the validation accuracy, which refers to the validation dataset in the three datasets used in this study. It is used to validate the learning results of this study. Finally, optimizers set the learning rate as mentioned earlier. We set the learning rate to 0.00025 instead of the common 0.0001 because the model cannot effectively learn at the initial stage.

We want to quickly move past that stage to avoid triggering the earlystop mechanism, and we do not want the model to learn poorly due to a high learning rate.

After adjusting the parameters, we then adjusted the model. To improve the model's performance, we added a BatchNormalization layer to normalize the data at the end of the model. We then used an activation function and a flattening layer to reduce the dimensionality of the model, followed by a Dropout layer to prevent overfitting. Finally, we used a Dense layer to classify the final result.

3 RESEARCH RESULTS

The final trained model's verification accuracy is shown in Table 3, where the EfficientNet B3 had the highest and best performance. When comparing EfficientNet with ResNet101, ResNet152, InceptionV3, and InceptionResNetV2, as shown in Table 4, the best-performing model is different from the one in [10] with ImageNet, and it turned out to be InceptionResNetV2 instead. The reason for this outcome may be related to the fact that the images used in this study were not commonly seen square-shaped pictures, and the random adjustments and data augmentation in the ImageDataGenerator might have had a slight impact.

Table 3: Comparison among EfficientNet versions

Module	Validation Accuracy	Testing Accuracy
EfficientNetB0	98.11%	96.32%
EfficientNetB1	98.32%	95.90%
EfficientNetB2	98.61%	95.90%
EfficientNetB3	98.49%	96.41%

Table 4: Difference among Models

Module	Validation Accuracy	Testing Accuracy
ResNet101	81.41%	91.48%
ResNet152	92.17%	83.13%
InceptionV3	96.98%	95.40%
InceptionResNetV2	98.70%	99.58%
EfficientNetB3	98.49%	96.41%

This study believes that the accuracy of recognition in stationary shooting and identification should reach at least 98% for possible substitution of quality control. Although this experiment encountered difficulties in data collection and experimentation due to relative data scarcity and excessively humid storage environments for masks, three models still achieved accuracy of 95% or more. This experiment focused on whether defects and normal masks can be accurately classified, so the confusion matrix of the test dataset was created as shown in Figures 6-8. It can be seen that in the case of InceptionV3 and EfficientB3, many masks with defects such as poor quality and wire flaws were classified as normal, and the most obvious ear loop breaks were also misclassified. This situation is absolutely unacceptable in practical applications, so adjustments may need to be made in terms of data or parameters. Although InceptionResNetV2 only misclassified five images, it may only have high accuracy for a part of this dataset, and further testing on the production line is needed to determine if its accuracy is truly high.

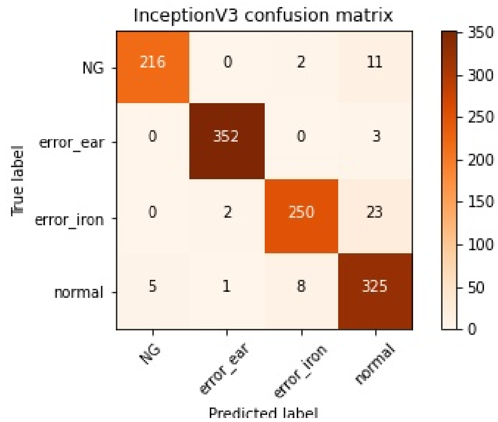

Figure 6: InceptionV3 Confusion Matrix.

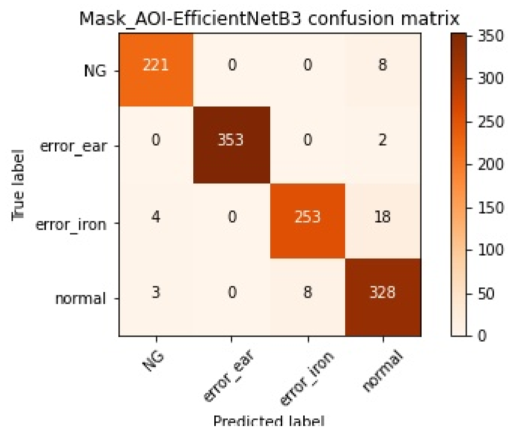

Figure 7: EfficientNetB3 Confusion Matrix.

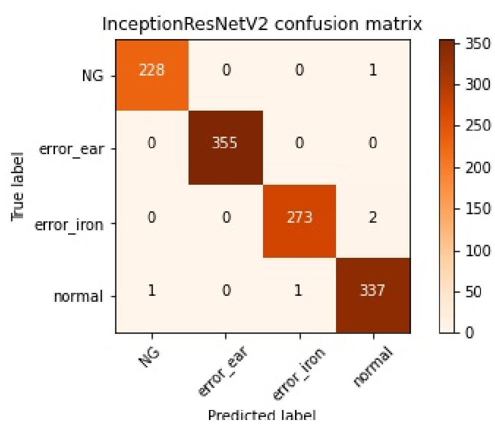

Figure 8: InceptionResNetV2 Confusion Matrix.

4 CONCLUSION AND FUTURE WORK

After conducting experiments, it was found that InceptionResNetV2 achieved the best performance on the test dataset. InceptionResNetV2 and EfficientNetB3 had a difference in accuracy of more than 3% on the test dataset, but their performance was not significantly different on the validation dataset. Although these differences may be related to the algorithms themselves, this study also used random data shuffling and data augmentation in the ImageDataGenerator, which may cause the model to have varying results in each learning iteration. Moreover, due to the equipment hardware limitation, half of the EfficientNet models were not used. These unused models such as B4 or above may lead to better results than InceptionResNetV2.

While using CNN for mask defect detection in a static state can achieve good accuracy, there are still cases where defective masks are misclassified as normal masks. Using CNN for mask defect detection on the production line still has a long way to go due to various challenges, such as:

1. Masks on the production line are constantly moving.
2. The camera may not be able to keep up with the speed of the moving masks, resulting in blurry images.
3. How to correctly detect masks from blurry images in the defect detection system.
4. How to activate the model and detect defects in the masks when they enter the camera's capture range.

In the future, object detection may be used as an auxiliary tool to help the defect detection system to perform detection more quickly. Additionally, using better cameras and more data may improve CNN's ability to accurately capture defects and improve the accuracy of the model. These will be the future research directions for our research team.

ACKNOWLEDGMENTS

Special thanks to Prof. Hsueh-Ting Chu from Asia University, Chairman Hsin-Ming Tseng from Yufa Technology Co., Ltd., and Jia-En Wang, who was a former master's student of Cheng-Yuan Ho at Asia University and currently works at Expetech Co., Ltd.

5 HISTORY DATES

Received June 2023; accepted July 2023

REFERENCES

[1] Bhuiyan, Md Rafiuzzaman, Sharun Akter Khushbu, and Md Sanzidul Islam. "A deep learning based assistive system to classify COVID-19 face mask for human safety with YOLOv3." 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT). IEEE, 2020.

[2] Yufa Technology Co., Ltd. http://www.pumma.com.tw/

[3] LeCun, Yann, et al. "Backpropagation applied to handwritten zip code recognition." Neural computation 1.4 (1989): 541-551.

[4] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521.7553 (2015): 436-444.

[5] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (1998): 2278-2324.

[6] Oh, Kyoung-Su, and Keechul Jung. "GPU implementation of neural networks." Pattern Recognition 37.6 (2004): 1311-1314.

[7] Bengio, Yoshua, Yann Lecun, and Geoffrey Hinton. "Deep learning for AI." Communications of the ACM 64.7 (2021): 58-65.

[8] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems 25 (2012).

[9] Hecht, Oded, and Giora Dishon. "Automatic optical inspection." 40th Conference Proceedings on Electronic Components and Technology. IEEE, 1990.

[10] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." International conference on machine learning. PMLR, 2019. Atul Adya, Paramvir Bahl, Jitendra Padhye, Alec Wolman, and Lidong Zhou. 2004. A multi-radio unification protocol for IEEE 802.11 wireless networks. In Proceedings of the IEEE 1st International Conference on Broadnets Networks (BroadNets'04). IEEE, Los Alamitos, CA, 210–217. https://doi.org/10.1109/BROADNETS.2004.8