Fashion-MNIST Classification Based on CNN Image Recognition

Zhengliang Chen [1,†], Mingrui Lv [2,*,†], Shiqi Tian [3,†], Shangyuan Yin [4,†]

[1] Victoria Hill School, Kunming, 650021, China

[2.]Stamford American International School, Singapore, 357684, Singapore

[3.]The High School Affiliated to Renmin University of China, Beijing, 100080, China

[4.]Tianjin Yinghua Experimental School, Tianjin, 301700, China

* Corresponding Author Email: 24mingruil@sais.edu.sg

[†]These authors contributed equally.

Abstract. Image classification is an important part of Artificial Intelligence (AI) and involves several tasks such as image normalization, image segmentation, feature extraction etc. Convolutional Neural Network (CNN) has been proved to be an effective network in image classification. According to this study, we had used a relatively small dataset named Fashion-MNIST (i.e. 70,000 images, 10 categories) to find out the model that can have a higher accuracy with limited training samples. We have trained three classic CNN models which are DenseNet121, MobileNet, ResNet 50 respectively. After that, we comprehensively measured the performance by several norms that comes from these three different models. Finally, we had a conclusion on which could be the most efficient one in this scenario based on the test results, in order to discovery which models are the most powerful one, and worth training. Human always deal with many types of images, so people need a powerful AI to help them to recognize and categorize these images. After this study, the DenseNet121 is the most powerful model in these three - DenseNet121, MobileNet, ResNet 50, the method to determine this result is that in the whole study we used a method called control variate method, we use the same amount of images, same amount of training times, then compared the final output of these three models, in the end we discovered that the DenseNet121 is the most powerful one.

Keywords: Neural Network, Image classification, Model comparison.

1. Introduction

Image classification is an important part of Artificial Intelligence (AI). It involves several tasks such as image normalization, image segmentation, feature extraction etc. Among all kinds of neural networks, Convolutional Neural Network (CNN) has been proved to be an effective network in image classification [1]. CNN is a type of deep learning neural network used to deal with structured data arrays for instance images, and it have been widely applied in computer vision field. It has become a cutting-edge technology in image classification, which has a series of models [2]. The basic inspiration is from how the visual cortex of human brain works when recognising objects. It was first discovered by David et al. in 1959 [3]. After this, in the 1980s, Kunihiko Fukushima designs an artificial neural network which is called Fukushima's Neocognitron, and the main idea is capturing complex patterns, using complex cells to absorb information from other lower-level cells. The Fukushima's Neocognitron is seen as the fundamental of Convolutional Neural Network, in 1990s, Yann LeCun et al. introduced the first modern application of CNN and developed LeNet-5 model [4][5]. LeCun's LeNet was trained on MNIST, and in their study Gradient-Based Learning Applied to Document Recognition, they provide the model with an example image then they ask the model to predict the label, after this, they update the model settings and make a comparison between prediction and real label value [6]. Finally, they repeat this process until the loss is minimized [4].

As the time passing by, CNN used more complex models train on larger datasets. With the development of powerful GPU machine, CNN can have a much better efficiency than before, in the past CNN can only discover simple and complex cells, but nowadays it can also do three-dimensional

object detection tasks and do very well in image classification [4]. In 2012, AlexNet, which is a deep CNN architecture which designed by Krizhevsky achieved a 16% error rate. Because of achievements of AlexNet, it has become an inspiration of different CNN models e.g. VGGNet, MobileNet and ResNet 50 [5].

Previous researchers have come up with a range of convolutional neural network models and related optimisation algorithms to get a better performance, yet it is hard to tell which approach is the best in specific usage scenario. In order to find out the networks that perform better than the others, people have designed many experiments to compare different CNN models, but in most cases, they have focused on comparing the performance of CNN models by using very large datasets. For instance, in 2018, Sultana et al. compared the performance of seven different CNN models on ImageNet dataset, which is a very large dataset that contains a total of 14,197,122 images with 1000 categories [5]. Dhillon et al. also proposed a comparison between various CNN models in 2019, they compared many classic models such as AlexNet, ZFNet, GoogleNet, etc [7]. In this paper, they used different datasets, usually large ones, and trained for an average of over a week to reach their conclusions.

In this study, we have used a relatively small dataset named Fashion-MNIST (i.e. 70,000 images, 10 categories) for the comparison, aiming to find out the model that can have a higher accuracy with limited training samples. We have trained three classic CNN models which are DenseNet121, MobileNet, ResNet 50 respectively. After that, we comprehensively measured the performance by several norms that comes from these three different models. Finally, we have had a conclusion on which could be the most efficient one in this scenario based on the test results. (depth, the effect of the parameters on performance)

2. Method

2.1. Dataset description

The dataset we used was Fashion-MNIST [8]. It contains a total of 70,000 images, each with a size of 28 × 28 pixels. In Fasion-MNIST, 60000 train images and 10000 test images exist. These images are chosen from 10 categories of fashion products, which are T-shirts, trousers, pullovers, dresses, coats, sandals, shirts, sneakers, bags and ankle boots, with 7, 000 different images in each category. The images in this dataset are all grayscale images. Figure 1 is some examples of the images in the category of dress in Fasion-MNIST dataset.

Figure 1. Examples of Fashion-MNIST images

2.2. Data preprocessing

After loading the dataset, we did some preprocessing on the images in order to adapt it to our work. According to the input requests of the models we chose, we resized all the images into squares with a size of 32 × 32 pixels, and then convert the grayscale images into RGB images by copying the single channel twice to form three-channel images.

Originally, pixel values were scattered in the range of 0 to 255. In order to speed up the convergence of training models and improve the comparability between different pixels, we normalized the images by dividing all the pixel values by 255, narrowing the pixel value range to 0~1. In addition, labels were encoded by using one-hot encoding.

2.3. CNN models

CNN model contains several layers, including convolutional layer, pooling layer and fully connected layer [9]. The function of convolutional layer is to extract features of pictures. By changing the convolutional kernel, model can extract different features. It can also boost performance by using summarized information. Then, the pooling layer is used to reduce the resolution and summarizes features generated by previous convolutional layer, thus will have a better performance. Fully connected layer is used in the end of CNN model, in order to summarize the previous extracted features and output the final prediction.

We chose three CNN models, including DenseNet121, ResNet50 and Mobile Net. ResNet50 has the largest amount of params, and less demanding on the capacity of memory. It is widely used in various purposes. DenseNet121 has much smaller number of params, which means it can perform much faster than ResNet50. Mobile Net has the smallest number of params among three models we chose.

Because the weight of the model is based on the ImageNet dataset, we have to make some changes on the layers of the model. Except from the base model, which is provided by TensorFlow, we also put another three layers, including dropout layer, global average layer and prediction layer. The dropout layer as well as global average layer are used to prevent overfitting of the model. The prediction layer outputs the final prediction with specified categories based on our dataset.

2.4. Implementation details

We chose to use Tensorflow to implement our training program and we used Apple M1 Pro GPU to run the program.

The loss function, optimizer, batch size and epochs were categorical crossentropy, Adam, 100 and 50, respectively. Moreover, our evaluation metrics included true positives, false positives, true negatives, false negatives, categorical accuracy, precision, recall and AUC.

3. Results and discussion

3.1. Performance of Mobile Net

After training the model with Mobile Net, we recorded the results and put them into excel to draw the charts to help us observe the results (Figures 2-4).

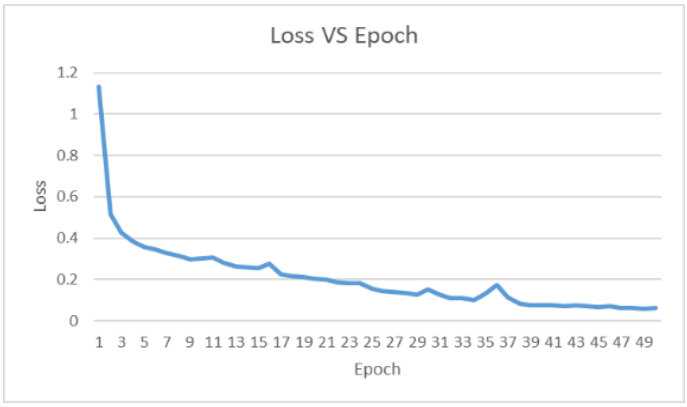

Figure 2. Loss change in Mobile Net

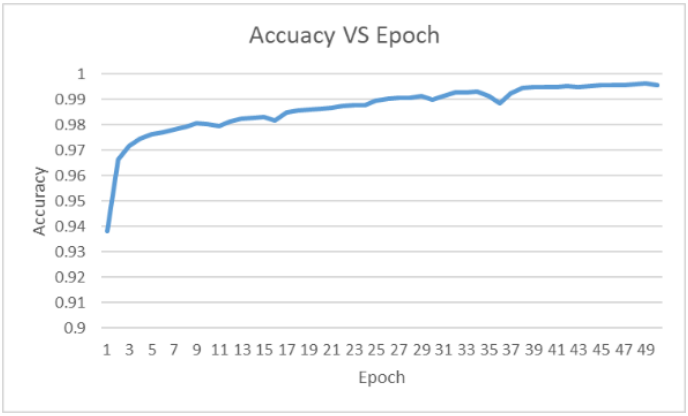

Figure 3. Accuracy change in Mobile Net

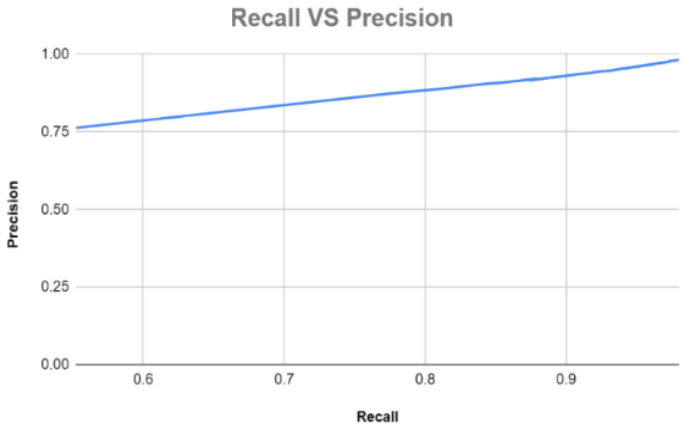

Figure 4. Precision per recall in Mobile Net

From the Loss-Epoch chart (Figure 2), it can be seen that the value of loss function is decreasing from nearly 1.1 to nearly 0 in 50 epochs. From the Figure 2, which is the Accuracy VS epoch chart, we can observe that the accuracy is overall increasing from approximately 0.93 to the end point 0.99 but decreasing in a few sections while epoch is increasing. In other cases, we can see the correlation of Recall-Precision chart is positive and it increased to a value near 1 from Figure 4. In addition, Table 1 presents the results of training.

Table 1. The final results of training

Final epoch	loss	accuracy	precision	recall	auc
Results	0.0619	0.9957	0.9802	0.9972	0.9992

3.2. Performance of DenseNet121

After training the model with DenseNet121, we recorded the results and put them into excel to draw the charts above to help us observe the results (Figures 5-7).

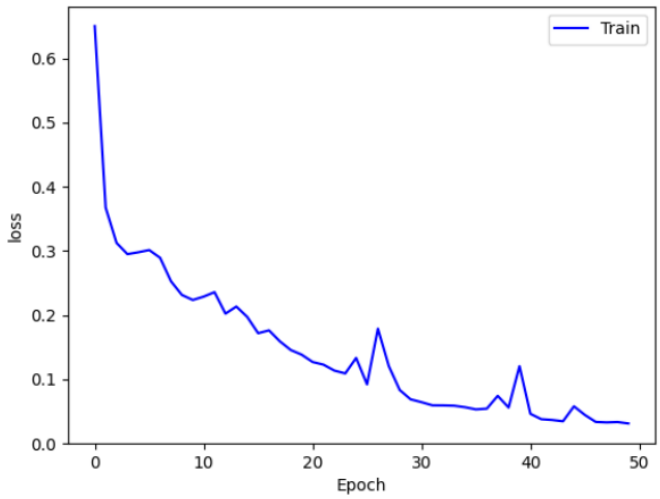

Figure 5. Loss change in Dense Net 121

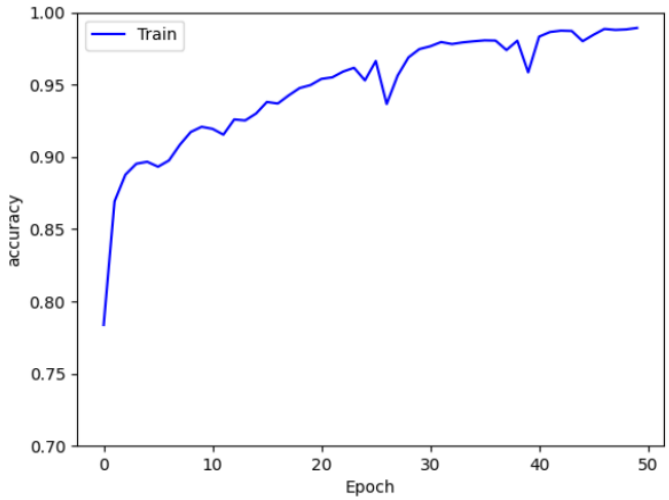

Figure 6. Accuracy change in Dense Net 121

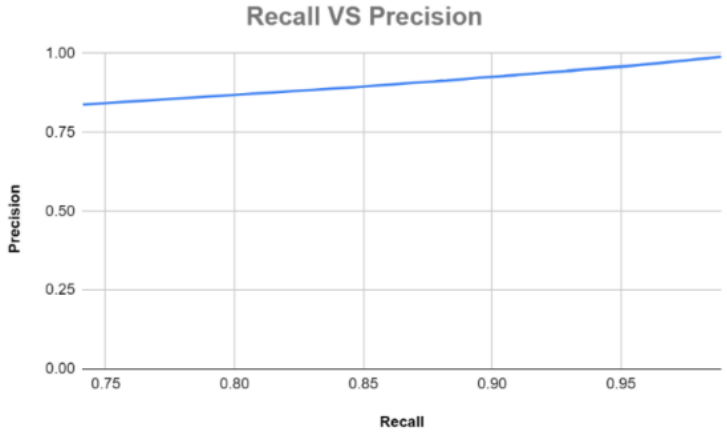

Figure 7. Precision per recall in Dense Net 121

From the Loss-Epoch chart (Figure 5), it can be seen that the value of the loss function is overall decreasing from nearly 0.65 to nearly 0 in 50 epochs and showed increases in the 27th, 46th and 40th epochs. From Figure 6, which is the Accuracy VS epoch chart, we can observe that the accuracy is

overall increasing from 0.79 to 0.99 but decreases in a few sections while the epoch is increasing. In addition, it can be seen that the precision and recall value have a positive linear correlation from the chart (Figure 7). So that we can observe that the precision of the model keeps increasing to 1.00 in 50 epochs while the recall of the model is increasing to 1.00. What is more, the final AUC of the model is 0.9996 (Table 2).

Table 2. The final results of training

Final epoch	loss	accuracy	precision	recall	auc
Results	0.0314	0.9893	0.9898	0.9889	0.9996

3.3. Performance of ResNet50

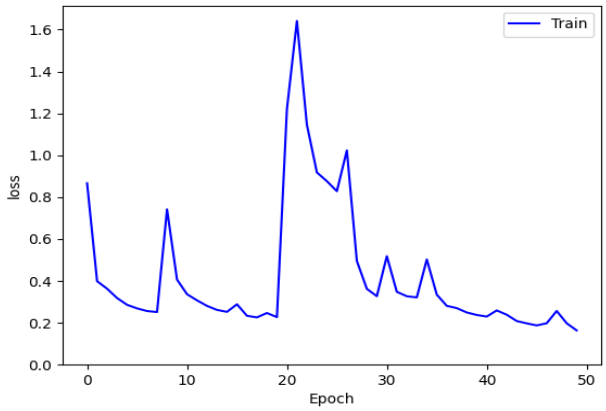

Figure 8. Loss change in Res Net 50

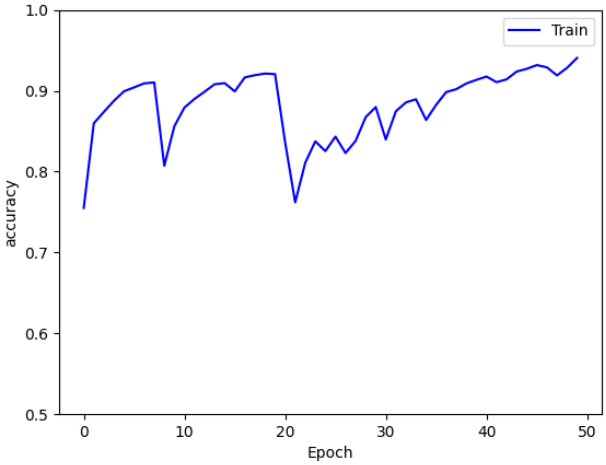

Figure 9. Accuracy change in Res Net 50

Table 3. Loss, precision, recall change in Res Net 50

	Before	Now
Loss	2.5731	0.2360
Precision	0.8185	0.9475
Recall	0.7024	0.9340

According to Figure 8, the Epoch and Loss of train, as the Epoch number rise from 0 to 10, the Loss rate first decline from approximately 2.6 to 0.7, then rise rapidly to 1.8. After this, the rest trend become stable as the Epoch number rising. the figure 9 illustrates a relationship between Epoch and accuracy, the whole trend could be seen as a rising trend, accuracy increasing from 0.70 to approximately 0.92. These two pictures all illustrate that using ResNet50, the Loss rate become lower, and the accuracy rate become higher, which means that the training efficiency rise as the epoch increases. The figure 10 shows changes of loss rate in the whole training process. The loss rate decline

from 2.5731 to 0.2360. which means that the whole training process is correct, and the learning efficiency is well. The total params is 23,608,202 and from figure 10, the trainable params is 23,555,082. The Non-trainable params is 53,120. As the number of precisions, it witnessed a rising trend, rose from 0.8185 to 0.9475, the number of recalls also had the same trend, increase from 0.7024 to 0.9340. Moreover, the final auc number is 0.9974. According to these numbers, they illustrate that the train process went pretty well, and ResNet50 is suitable for processing image classification, but this can be done better if larger dataset is used, and more powerful GPU to support more training times.

4. Conclusions

In this work, we proposed a comparison between three CNN models using Fashion-MNIST dataset. We used loss function, accuracy and precision-recall diagram to evaluate the performance of the three models. Experimental results showed that the Dense Net 121 model trained with our method performed better than the other two models. That's because Dense Net 121 is a lightweight model. It requires less parameters and computations to achieve the similar accuracy compared with Res Net 50. Although Mobile Net is also a lightweight model as Dense Net 121, the latter has a strong resistance to overfitting, which makes it particularly suitable for poor data trainings.

However, we only chose three classic models for the comparison. The advantage of the Dense Net 121 model wasn't clear enough, so we still can't conclude that this model was the most suitable choice for small dataset training. In addition to this, we didn't use a very powerful GPU for the training, so we were unable to test any complex models with a large amount of parameters, such as VGG Net. In the future, we plan to use a more powerful GPU to test more CNN models including some new released models and get a more complete comparison between all the CNN models. We also plan to increase the number of experiments and average the results of each experiment to get a more reliable result.

References

[1] O'Shea K. et al. An introduction to convolutional neural networks [R]. arXiv preprint arXiv:1511.08458 (2015).

[2] Wood T. Convolutional Neural Network [R]. DeepAI, 17 May 2019, https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network

[3] Raschka S et al. Python Machine Learning: Machine Learning and Deep Learning with Python, Scikit-Learn, and Tensorflow 2 [R]. Packt, 2019.

[4] Yalçın O G. The Brief History of Convolutional Neural Networks [J]. Medium, Towards Data Science, 24 Feb. 2021,

[5] Sultana F et al. Advancements in image classification using convolutional neural network [C]. 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN). IEEE, 2018.

[6] LeCun Y. Gradient-Based Learning Applied to Document Recognition [R]. IEEE Xplore, Nov. 1998

[7] Dhillon A et al. Convolutional neural network: a review of models, methodologies and applications to object detection [J]. Progress in Artificial Intelligence 9.2 (2020): 85-112.

[8] Xiao H et al. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms [R]. arXiv preprint arXiv:1708.07747 (2017).

[9] Khan S et al. A guide to convolutional neural networks for computer vision [J]. Synthesis lectures on computer vision8.1 (2018): 1-207.