COMP4471 Project: Gundam Classifier with Transfer Learning

Ho Chun To
20533455
cthoab@connect.ust.hk

HE Jianle
20536134
jhebg@connect.ust.hk

1. Abstract

"Mobile Suit Gundam" is a famous series in animation. In this project, we proposed to apply transfer learning with a pretrained VGG19 with ImageNet dataset, to classify the name and model of Gundam. The most time consuming part in this project is data scraping, there is not structured dataset available on the Internet. Therefore, we have to look for appropriate image dataset from image sharing platform, personal blog and even searching engine. So that we have to perform data cleaning manually, to remove all inappropriate images. Due to limited time, our final result perform not as good. There is a large gap between training loss and validation loss, the model overfitted. This is what we expected, the quantity and quality of training dataset are not good enough, we will discuss more in Data section.

2. Introduction

Classification problems are popular in the area of machine learning, there is a diversity in the classification models. However, the quantity of classification models related to animation is much fewer. Developing a machine learning models for the animations is base of the love of the fans. Meanwhile we find that there is a challenging task for the a animation that we love. It is very hard for people to recognize all the models of "Mobile Suit" in the whole series of the "Mobile Suit Gundam" animations. Only die heart fan of this series would have the ability to classify all of the machines. Is it possible to train a classification model having the same ability compare to those fans? There are two approaches for tackling this problem. Our first approach is to build a convolutional neural network with custom architecture for the image classification and the second approach is to do some transfer learning with ImageNet model to evaluate the result. Numbers of image processing algorithms will also be implemented, including histogram of color (HoS), gradient and orientation histogram, and some data augmentation algorithms. The final result is not as good, but this fulfil our evaluation at the beginning of the project.

3. Related Work

There are some related project for anime character labeling, which shows the possibility to perform an image classifier to classify different kinds of Gundam(s) through the Convolutions Neural Network model.

For the dataset, due to they are image labelling problems. The training dataset are image with position and class labels. But we are a singe image classification, so that our data are photo of Gundam toy model (Gunpla).

For the model achitecture, we are using the pretrained VGG19, but the related project construct a shallow CNN Model with a few convolution layers and a few fully connected layers.

4. Data

To achieve the research goal, a large dataset of Gundam images and their name are needed for the image classification. However, there are no pre-processed dataset of Gundam available online, it is decided to collect the images of Gundam from the Internet by web scraping. In this state, there are three different data sources, "crazypipogunpla" and "suparrobo", the personal blogs and grabbing image from Google image. Both of the data sources provide the Gunpla product images with different kinds of angle views with mostly black background with large resolution. However, due to the limited image collected from online, data augmentation technique is applied for increasing the number of training data to feed into the model. For instance, applying flipping, zooming in/out, shearing the image can increase the number of sample data.

The challenging part of web crawling is that, the searching function of personal blog or photo sharing platform are not reliable. We are searching the images with the model name that previously scraped from fan-made wikipedia of Gundam. When the name is too general, there are a lot irrelevant image been downloaded. But when the name is too specific, there will be no searching result from the platform. At the end we crawl as much as image and perform data cleaning manually. There are a huge amount of Gundam models in the series, the first time of crawling found that

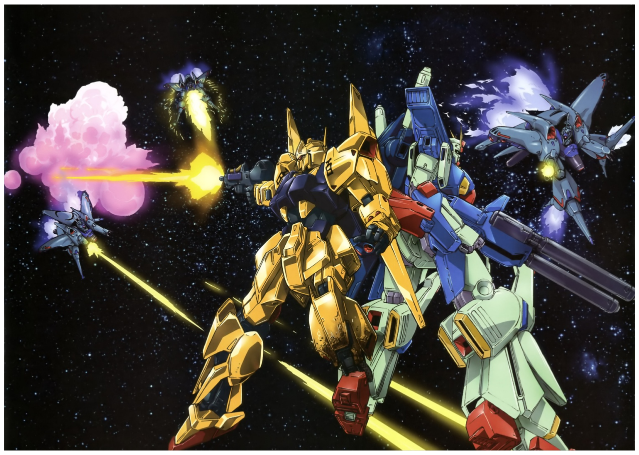
Figure 1. Polluted data multi kinds of Gundams in the same image

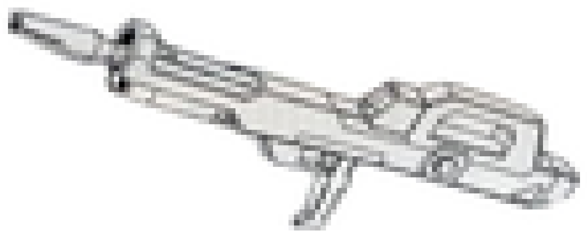
Figure 2. Polluted data multi kinds of Gundams in the unrelated image

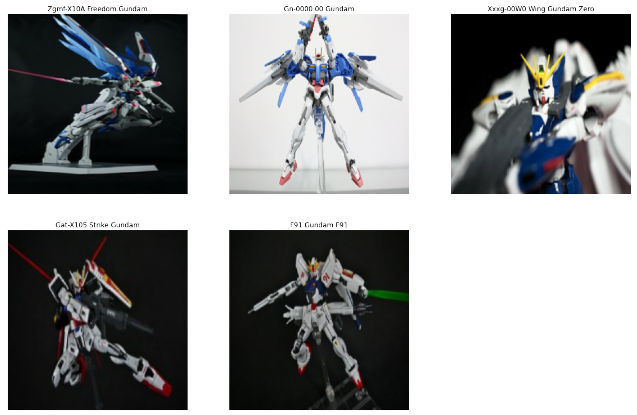

Figure 3. Resized image

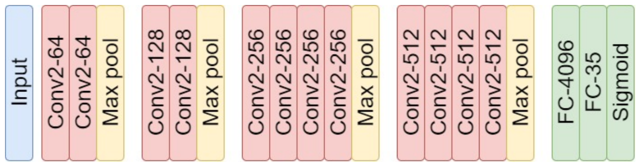

Figure 4. customized vgg19 model

Hyperparameter	Value
No. of epochs	30
Learning Rate	0.01
Iterations	40

Table 1. The default hyperparameters setting

there are over 900 classes. Most of the models are not famous and there are extremely lack of image. So we decided to remove those classes with limited image. We keep the most popular model in our development, so the final number of classes is around 40.

For the data cleaning, since there are some images crawl from the internet that are not related to the image class itself, such as the button images, blogger banners, advertisement logos, multi Gunpla images,wrong Gunpla pictures as figure 1, 2 shows. These pictures will be deleted manually to reduce the pollution of the image dataset .

For the data pre-processing, there are images in different resolution size, in order to fit the network input of the model, resizing technique are used to reduce the image resolution from their original size into (224, 224). It means that the images are converted into square shape to fit with the imput size of our transfer learning input. Some sample of the resized images shown as figure 3 below.

After the data filtering, there are 1529 of training images and 363 validation images from 35 Gundam classes, 131 images are used as test image.

5. Methods

In order to perform image classification, a convolutional neural network are used. However, due to our number of data in training set is too small for the network to train, resulting with a very low accuracy. Also, with the limited computational resources, it is not efficient to train with 22,269,027 each step, resulting that the training speed of each epochs is extremely slow.

In order to solve the problem, transfer learning with a pretrained VGG19 model with imageNet is used as our model. We keeps the convolutions layers of the VGG19 model and customize the fully connected layers from two FC-4096, one FC-1000 with softmax layer to one FC-4096, FC-35 with sigmoid layer to reduce the model's complexity.

During the training process, the first 20 layers' parameters of our model are frozen to keep the pre-trained model's weight and fine tune the Conv2-512, FC-4096, FC-1000 layers in the model. The customized model architecture is shown below as figure 4 as our baseline model.

We further fine tune the model to unfreeze one more Conv2-512 layer in the model.

6. Experiments

The environment of the experiments are run on Google Colab and the hyperparameter setting is shown below.

We had trained with the base model only as the default hyperparameters. As the graph 5 as shown. It shows that

Hyperparameter	Value
No. of epochs	20
No. of epochs of 1st fine tune	20
No. of epochs of 2nd fine tune	20
Learning Rate	0.01
Iterations	40

Table 2. The first fine tune hyperparameters setting

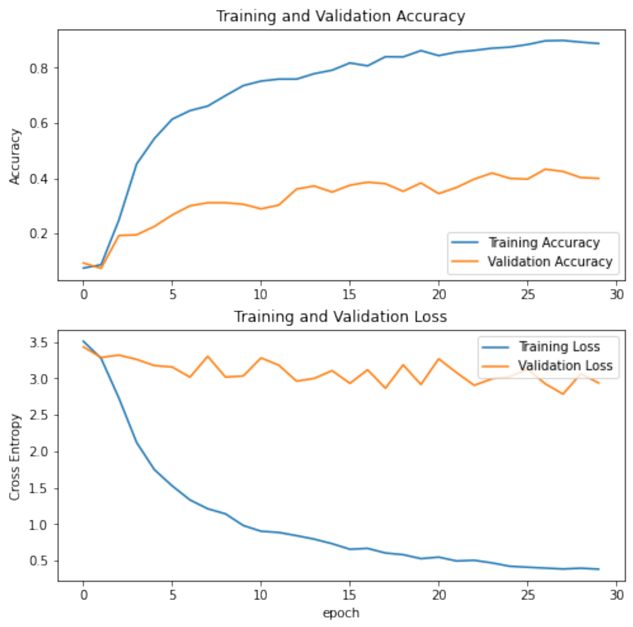

Figure 5. the training and validation curve and the loss curve of default hyperparameter setting

there is a gap between training set and validation set in the training accuracy and loss. And the resulting test accuracy with about 62 percent. It means that overfitting occurs while the training process, therefore, we fine tune the model in order to get a more accurate result in test set.

We had done some experiment on how the fine tune affecting the loss and the accuracy. We did two setting of changing the number of epochs on baseline model, first fine tune model and also second fine tune model. For the 1st fine tune hyperparameter setting are shown below as table 2.

The fine tuning accuracy curve and the loss curve for the model are shown as figure 6. The first 20 epochs are the original setting of the model, 21 to 40 epochs are the first fine tuning and the 41 to 60 epochs are the second fine tuning. We can observe that after both fine tuning, the loss drops in a steady rate and the accuracy increases. Meaning that these two fine tuning step can improve our training quality of the model. This fine tune boost the training accuracy into 78 percent and the testing accuracy 68 percent.

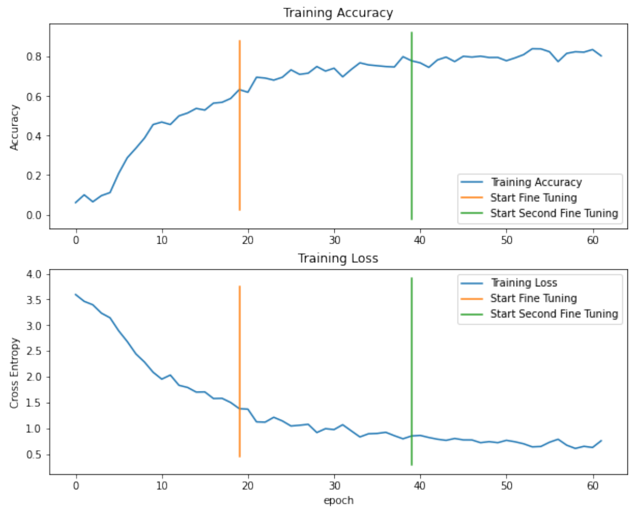

Figure 6. the training curve and the loss curve of default hyperparameter setting

7. Conclusion

To sum up, our method from section 4 successfully to classify the name of the Gundam from the image. Fine tuning the model could help increasing the test accuracy to classify the image. Due to the limited time, the accuracy of the model is not very high. It is believed that it leaves rooms for improvement for fine tuning in order to have better classification results.