2022/1

TAKE-HOME MIDTERM EXAM

2110531 DATA SCIENCE AND DATA ENGINEERING TOOLS

Traffy Fondue

PREPARED FOR

Aj. Peerapon Vateekul

PREPARED BY

6570146121
Pongsakorn Tikapichart

TABLE OF CONTENTS

Ponsakorn Tikapichart
6570146121

Chapter 1: Introduction

This assignment is the individual take-home midterm examination for the 2210531 Data Science and Data Engineering Tools course at Chulalongkorn University's Faculty of Engineering. This project's objective is to identify the best image classification model in the Traffy Fondue Dataset.

Traffy Fondue is an application that collects trouble reports. Suggestions from the informant and the support system for better problem management In which the informant is not required to know the staff or who was previously responsible for the situation. The system is intended to be simple to use. Only the informant takes a photograph and indicates the nature of the problem. The system will quickly notify the staff and the responsible team of the concern.

Chapter 2: Data preparation

The data is separated into two sets: the training set (33,143 images) and the testing set(6,425 images).

The training data is divided into ten categories: flooding, electric, canal, stray, light, traffic, road, sidewalk, sanitary, and sewer. However, because certain photographs were incorrectly classified into subfolders, I manually eliminated some irrelevant images from each class. As a result, the training data was reduced from 33,143 to 20,886 pictures. The figure below depicts training data before and after the cleaning processes in each subcategory.

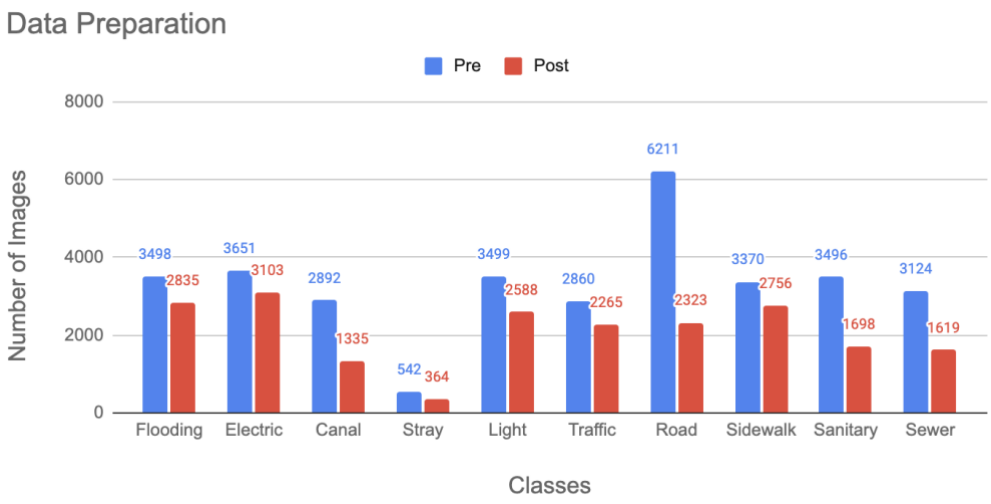


Figure 1: Training data before and after cleaning process in subcategories

The test set data has already been cleansed. However, when I import the data into Google Colaboratory, the name of test files is in the following order: 0.jpg, 1000.jpg, 1001.jpg,.....,998.jpg, 999.jpg. As a result, the order of the testing data is incorrect when generating the csv file. Therefore, I altered the names of the testing image files to 0000.jpg, 0001.jpg, 0002.jpg,.... 6424.jpg using Python code as follows

```python
import os
path = '/Users/topp/Downloads/ds-tools/TraffyFondue/test-rename'
for filename in os.listdir(path):
    name = filename[:-4]
    suffix = filename[-4:]
    prefix = name[:4]
    num = name[4:]
    num = num.zfill(4)
    new_filename = prefix + num + suffix
    os.rename(os.path.join(path, filename), os.path.join(path, new_filename))
```

Figure 2: Python code for changing test file name from test0.jpg to test0000.jpg

Then I moved all of the image test files into a subfolder so that I could compile with the 'flow from directory' function in TensorFlow Library.

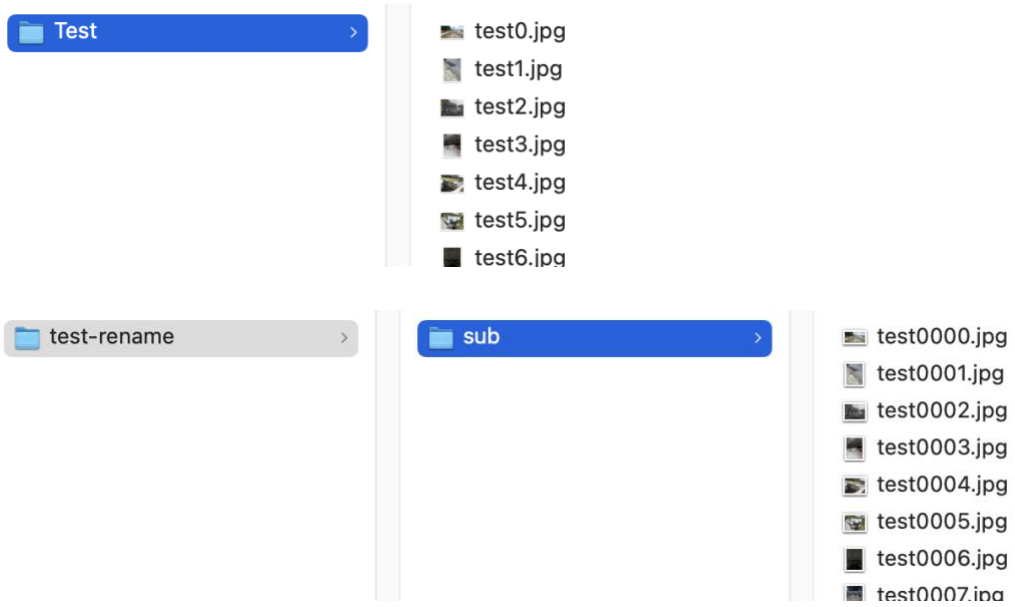


Figure 3: Result from running python code

Then, zip both the cleaned training data and the testing data and upload them to Google Cloud. As following urls.

training_set_url = "https://storage.googleapis.com/data-science-tools-2/train-filtered.zip"
Testing_set_url = "https://storage.googleapis.com/data-science-tools-2/test-rename.zip"

Chapter 3: Model

I ran five experiments, and my best model is EffcientNetV2M with flatten, dropout, and dense layers, as seen in the figure:

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 efficientnetv2-m (Functiona  (None, 10, 10, 1280)     53150388
 l)

 flatten_5 (Flatten)         (None, 128000)            0

 dropout_12 (Dropout)        (None, 128000)            0

 dense_12 (Dense)            (None, 256)               32768256

 dropout_13 (Dropout)        (None, 256)               0

 dense_13 (Dense)            (None, 10)                2570

=================================================================
Total params: 85,921,214
Trainable params: 85,629,182
Non-trainable params: 292,032
_________________________________________________________________
```

Figure 4: Final model

The training parameters are shown in the table below:

Table 1: Training parameters

Parameters	Values
Image size	300 * 300
Batch	16
Pretrain dataset	ImageNet
Learning rate	0.0001
Loss	categorical_crossentropy
Reduce learning rate on plateau	Patience = 3
Iterations	20

Image augmentation parameters are as below

Table 2: Augmentation parameters

Parameters	Values
Shear range	0.2
Zoom range	0.2
Flip	Horizontal only
Fill mode	Nearest

Chapter 4: Results

The accuracy from the validation data set is 0.79089 after 20 epochs of running the model.

```
Epoch 20/20
1045/1045 [==============================] - ETA: 0s - loss: 0.0382 - accuracy: 0.9841
Epoch 20: val_accuracy improved from 0.78777 to 0.79089, saving model to efnv2m_best_model.h5
1045/1045 [==============================] - 936s 895ms/step - loss: 0.0382 - accuracy: 0.9841 - val_loss: 1.6398 - val_accuracy: 0.7909 - lr: 1.0000e-05
```

Figure 5: Result after running 20 epochs

The chart below illustrates the accuracy and loss for each epoch for training and cross validation data sets.

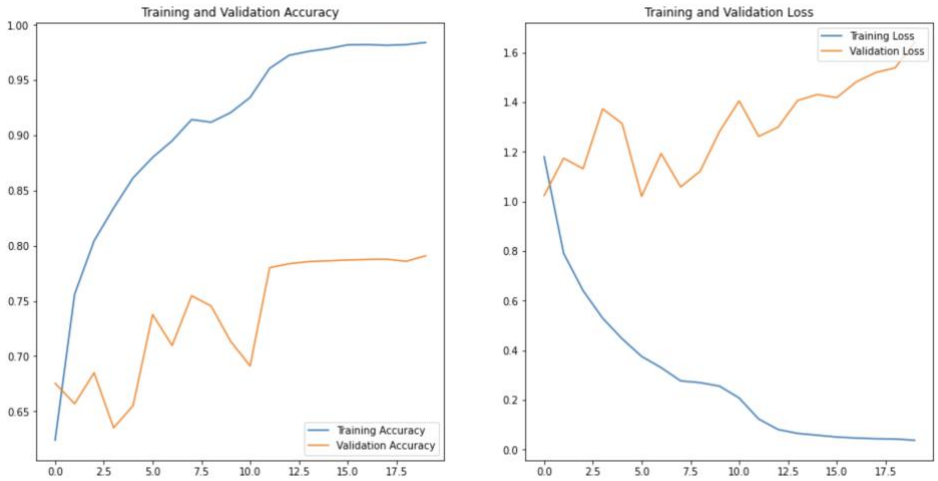
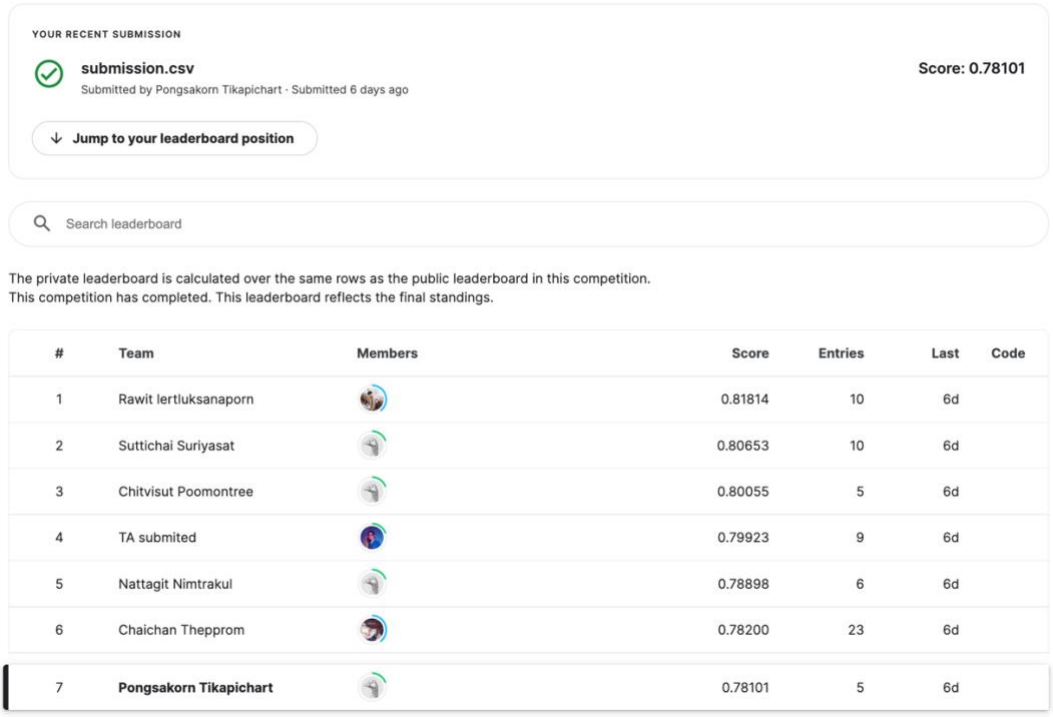


Figure 6: the accuracy and loss for each epoch for training and cross validation data

The outcome of Kaggel testing data is 0.78101, which places 7th out of 39 contestants.



#	Team	Members	Score	Entries	Last	Code
1	Rawit Iertluksanaporn		0.81814	10	6d	
2	Suttichai Suriyasat		0.80653	10	6d	
3	Chitvisut Poomontree		0.80055	5	6d	
4	TA submited		0.79923	9	6d	
5	Nattagit Nimtrakul		0.78898	6	6d	
6	Chaichan Thepprom		0.78200	23	6d	
7	**Pongsakorn Tikapichart**		0.78101	5	6d	

Figure 7: Outcome from Kaggle

Chapter 5: Discussion

Diagnose the models

I have run 5 experiments various parameters as below

Table 3: Experiment training parameters

No. of experiments	1	2	3	4	5
Based model	EfficientNet V2B1	EfficientNet V2M	EfficientNet V2L	EfficientNet V2L	EfficientNet V2M
Image size	240*240	240*240	240*240	240*240	300*300
batch	24	24	24	16	16
Epoch	10	10	20	20	20
Test results on kaggle	0.73959	0.76253	0.77266	0.77740	0.78101

I started the experiment with EfficientNetV2B1 since, from my research, EfficienctNetV2 is the state of the art for image classification. And the reason I chose a smaller size model at first is that I want to try to submit the outcome to Kaggle as soon as possible to check whether any error codes need to be addressed.

The model was then scaled up to M and L with the same parameters so that I could compare the results. I discovered that increasing the Epoch and Image size improves the outcome.

Next steps for improvement

The next experiment that could produce better results is 'EfficientNetV2L' model with increasing the image size to 480*480. I also attempt an additional Epoch, which could be 100 and setting early stop when the accuracy is not increased for 10 epochs. I will try to experiment with altering the batch size to 32 and different batch sizes to see how well the model predicts. In addition, I would add more layers on top. I'd also like to try switching the pretrain picture from ImageNet to ImageNet-21K, which the research claims have higher accuracy than the ImageNet pretrain dataset. Furthermore, I would add more data into training data set by adding removing pictures into the correct categories.

Chapter 6: Conclusion

This report describes the 'EfficientNetV2M' image classification model on the Traffy Fondue dataset in detail, including the EffiencientNetV2M structure, parameter settings, and data set pre-processing. The accuracy on the test data set is 0.78101, ranking 7th out of 39 in the Kaggle competition.