Homework 4 Report for -

ME 592

Data Analytics and Machine Learning for Cyber-Physical Systems Applications

Spring 2022

Group Theme - Agriculture

Submitted By

Md Sakib Ferdous
Chemical and Biological Engineering Department

Ibne Farabi Shihab
Computer Science Department

Iowa State University

Submitted To

Dr Soumik Sarkar

Course Teacher

&

Xian Yeow Lee

Course TA

Answer to the Question no. 1

The following steps are carried out as indicated in the question.

Initial model

1. Loaded the dataset with code block –

```
### Preparing Datasets and Dataloaders for Training

# Dataset Object
train_dataset = GWD(train_df, DIR_TRAIN, mode = "train", transforms = T.Compose([Augmenter(), Normalizer(), Resi
valid_dataset = GWD(valid_df, DIR_TRAIN, mode = "valid", transforms = T.Compose([Normalizer(), Resizer()]))

# DataLoaders
train_data_loader = DataLoader(
    train_dataset,
    batch_size = 8,
    shuffle = True,
    num_workers = 4,
    collate_fn = collater
)

valid_data_loader = DataLoader(
    valid_dataset,
    batch_size = 8,
    shuffle = True,
    num_workers = 4,
    collate_fn = collater
)


test_data_loader = DataLoader(
    valid_dataset,
    batch_size = 1,
    shuffle = True,
    num_workers = 4,
    collate_fn = collater
)
```

RetinaNet models

2. The data was split into train (80% of data) and test (20% of data).

```
### Splitting Train Dataset into train - val (80:20)

images = df['image_id'].unique()
valid_imgs = images[-674:]
train_imgs = images[:-674]

valid_df = df[df['image_id'].isin(valid_imgs)]
train_df = df[df['image_id'].isin(train_imgs)]
```

3. Model parameters were –
    a. Batch size 8 with
    b. 4 workers
    c. 15 epochs

    We did not try any early stopping.

4. Throughout the homework we used the same model structure taken from GitHub given in the question.

5.Results:

	Classification Loss	Regression Loss	Running loss
Train	0.116	0.436	0.863
Val	0.155	0.480	0.659
Test	0.163	0.432	0.651

Answer to the Question no. 2

In this task we tried to improve the model by doing hyper parameters tuning by doing the following(We tried a bunch. We are reporting the best 5 and epochs were variable length due to time constraint)

a) Different backbone model
b) Variation in optimizers and learning rates
c) Different loss function

Model name	Epochs	Learning rate	Optimizer	Classification Loss	Regression Loss	Classification Loss	Regression Loss
Resnet 34	3	0.0001	SGD	0.166	0.504	0.119	0.417
Resnet 50	15	0.0001	Adam	0.094	0.351	0.174	0.449
Resnet(2:1loss) 18	2	0.0001	Adam	0.193	0.600	0.156	0.485
Resnet 18	2	0.0002	SGD	0.182	0.555	0.223	0.573
Resnet 50	5	0.0001	Adam	0.114	0.397	0.160	0.598

Notes about the models:

1. We tried resnet 50, 34 and 18 as backbone networks
2. Resnet 50 provided the lowest training loss while Resnet 34 provided the lowest validation error
3. SGD optimizers seems to work better than Adam
4. During our experiments we found that higher learning rates produced inaccurate results.
5. 2:1 loss Only give a good result among the few we tried.

Answer to the Question no. 3

- ➔ Retina net is basically a one-staged object detection model. It can take two networks; one is unified network composed of a backbone network and two task-specific subnetworks. Backbone is based on existing network, and it is responsible for computing a conv feature map over a particular image. Among the subnets the first do classification on backbones output while the later one does convolutional bounding box.
- ➔ RCNN needs huge time to train as it needs a lot of pre-classified regions while Retina net does bounding box prediction without any region proposal. To be more specific, Faster RCNN and Fast-RCNN are two staged detectors and they use region proposal technique to predict the bounding boxes which make them complicated while use pooling layer for next regression and classification. We can say in a sentence that one stage detectors(Retina net) have high inference speed while two-stage detectors have higher accuracy in terms of localization and recognition.

Code link: https://github.com/farabi1038/ME-592/tree/main/HW3

Answer to the Question no. 4

Using Model to Annotate Canopy Image

For this part, we did the following steps:

1. Crop the images to find leaf and save them using some tools
2. Load the best pkt model from question 3 with the structure.
3. Do predictions on those images using our model

Below are the prediction for the images from our model(Ignore the number 2 after every image name):

```
Using: cpu
name of image :1006 (3).jpg prediction: 1
name of image :1007 (2).jpg prediction: 1
name of image :1006 (2).jpg prediction: 1
name of image :1004 (2).jpg prediction: 4
name of image :1001_C (2).jpg prediction: 1
name of image :1005 (2).jpg prediction: 4
name of image :1009 (2).jpg prediction: 4
name of image :1001 (3).jpg prediction: 6
name of image :1001 (2).jpg prediction: 4
name of image :1008 (2).jpg prediction: 1
name of image :1003 (2).jpg prediction: 4
name of image :1002 (2).jpg prediction: 4
name of image :1005_C (2).jpg prediction: 1
```

The strategy for automating the process