A Reprot for Deep Learning Lab Exercise 6

Ziyi Guo, MSc Data Science

zg2u21@sonton.ac.uk

1 FineTuning ResNet50 in Transfer Learning

In the first part of the experiment, transfer learning models applying and fine tuning ResNet50 are implemented and tested as below.

```
model = resnet50(pretrained=True)
model.avgpool = nn.AdaptiveAvgPool2d((1,1))             # Tuning Pooling Layer
model.fc = nn.Linear(2048, len(train_dataset.classes))  # Tuning Linear Output Layer
model.train()

for param in model.parameters():       # Freezing layers by not tracking gradients
    param.requires_grad = False
model.fc.weight.requires_grad = True   # Unfreezing last layer weights
model.fc.bias.requires_grad = True     # Unfreezing last layer biases
optimiser = optim.Adam(filter(lambda p: p.requires_grad, model.parameters()), lr=1e-4) # Defining Optimiser and Optimising non-frozen layers
loss_function = nn.CrossEntropyLoss()
device = "cuda:0" if torch.cuda.is_available() else "cpu"
trial = Trial(model, optimiser, loss_function, metrics=['loss', 'accuracy']).to(device)
trial.with_generators(train_loader,val_generator=val_loader,test_generator=test_loader)
trial.run(epochs=100)
results = trial.evaluate(data_key=torchbearer.VALIDATION_DATA)

predictions = trial.predict()         # Test and Report Model Performance
predicted_classes = predictions.argmax(1).cpu()
true_classes = list(x for (_,x) in test_dataset.samples)
print(metrics.classification_report(true_classes, predicted_classes, target_names=train_dataset.classes))
```

With a pre-trained ResNet50 model, changes are made to both the pooling layer from Global Average Pooling to the Adaptive Average Pooling and the output layer from default linear output to fitting the class amount of the training data, which adapt the original model to the non-square inputs and number of labels of the dataset. The new model is named ResNet50-1. Further, to improve the performance, attempts are made to alter the learning rate. With different learning rates of 0.01 and 0.000001, models based on ResNet50-1 are built (ResNet50-2, ResNet50-3) and tested as well.

	Precision				Recall				F1-Score				Support			
	CNN	ResNet-1	ResNet-2	ResNet-3	CNN	ResNet-1	ResNet-2	ResNet-3	CNN	ResNet-1	ResNet-2	ResNet-3	CNN	ResNet-1	ResNet-2	ResNet-3
Alilaguna	0.63	0.53	0.36	0.00	0.63	0.53	0.47	0.00	0.63	0.53	0.41	0.00		19		
Ambulanza	0.25	0.50	0.35	0.00	0.14	0.55	0.32	0.00	0.18	0.52	0.33	0.00		22		
Barchino	0.00	0.62	0.08	0.00	0.00	0.16	0.04	0.00	0.00	0.25	0.05	0.00		51		
Gondola	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		3		
Lanciafino10m	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		7		
Motobarca	0.05	0.11	0.12	0.00	0.02	0.05	0.03	0.00	0.03	0.07	0.05	0.00		59		
Motopontonerett	0.00	0.50	0.50	0.00	0.00	0.67	0.33	0.00	0.00	0.57	0.40	0.00		3		
MotoscafoACTV	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		1		
Mototopo	0.66	0.69	0.75	0.29	0.91	0.85	0.70	0.62	0.76	0.76	0.73	0.39		274		
Patanella	0.34	0.35	0.26	0.00	0.70	0.53	0.69	0.00	0.46	0.42	0.37	0.00		74		
Polizia	0.00	0.44	0.13	0.00	0.00	0.27	0.27	0.00	0.00	0.33	0.17	0.00		15		
Raccoltarifiuti	0.00	0.50	0.36	0.00	0.00	0.58	0.68	0.00	0.00	0.54	0.47	0.00		19		
Sandoloaremi	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		3		
Topa	0.00	0.17	0.33	0.00	0.00	0.03	0.07	0.00	0.00	0.06	0.11	0.00		29		
VaporettoACTV	0.97	0.97	0.98	0.58	0.98	1.00	0.97	0.89	0.98	0.98	0.98	0.70		325		
Water	0.97	0.95	0.96	0.76	0.95	0.93	0.87	0.41	0.96	0.94	0.91	0.53		420		

Table 1. Per-class Precision and Recall Values of Classification Models on Test Set

	CNN	ResNet50-1 (lr=1e-4)	ResNet50-2 (lr=1e-2)	ResNet50-3 (lr=1e-6)
Accuracy_Train	0.747	0.888	0.979	0.460
Accuracy_Valid	0.701	0.754	0.722	0.423
Accuracy_Test	0.766	0.783	0.727	0.477

Table 2. Training, Validation and Test Accuracy of Classification Models

It can be observed that the CNN model in the previous experiment achieves the training and test accuracy of 0.747 and 0.766, while the ResNet50 achieves the training and test accuracy of 0.888 and 0.783 and generally performs better. Altering the learning rate of the optimisers, the ResNet50-2 with learning rate of 0.01 achieves a set of greater training but lower test accuracy of 0.979 and 0.727, which seems to be overfitting; while the ResNet50-3 with learning rate of 0.000001 achieves the training and test accuracy of 0.460 and 0.477 which seems to be underfitting and not fully learned.

2 Feature-based SVM classifier

In the second part of the experiment, an SVM classifier is trained based on the features extracted from the network and tested as below.

```
clf = sklearn.svm.SVC(gamma='scale').fit(training_features, training_labels)
score_train = clf.score(training_features, training_labels))
score_valid = clf.score(valid_features, valid_labels))
score_test  = clf.score(testing_features, testing_labels))
accuracy_class = f1_score(testing_labels, clf.predict(testing_features))
```

	CNN	ResNet50	SVM
Accuracy_Train	0.747	0.888	0.878
Accuracy_Valid	0.701	0.754	0.825
Accuracy_Test	0.766	0.783	0.872

Table 3. Training, Validation and Test Accuracy of Classification Models

It can be observed the feature-based SVM model has similar training accuracy to the ResNet50 model, but achieves overall higher validation and test accuracy than the ResNet and is not overfitting inferred from the similar training, validation and test accuracy. Detailed study on the test accuracy of the SVM model in each class shows similar 0 and quite low accuracies for the classes ***Gondola***, ***Lanciafino10m***, ***MotoscafoACTV***, etc, but higher accuracies in other classes, indicating the cause of low classification accuracy to be the lack of data. As for training speed, it takes around 6 seconds for the SVM to train, while it takes around 12 seconds in each iteration and 20 minutes in all for the ResNet50 to train. The training of feature-based SVM is much faster than the ResNet. In conclusion, according to the model performance and operation speed, the feature-based SVM is much better than the ResNet50 model.