COL774 – Machine Learning

Assignment 4

Nilaksh Agarwal
2015PH10813

Part 1 – Fixed Algorithms:

Part (a) – SVM:

Libraries Used:

- OpenCV
- PILLOW
- Pickle
- Sklearn

Preprocessing:

- Converted to grayscale. *PIL.Image.open(file).convert("L")*
- Cropped the image, taking only [33:194, 8:152] to get rid of border and score box
- Downscaled the image by 2, using [::2, ::2] to take every alternate row and column
- Also specified the datatype of image array as uint8 to save space

HyperParameter Training:

- Varying C (the penalty term) from [0.1, 0.3, 1.0, 3.0, 10.0], we observed for smaller C [0.1, 0.3] the model predicted just the majority class. Since it was a skewed dataset, it obtained almost 97% accuracy with this. For a larger C, the SVC's accuracy decreased, since it was now misclassifying some majority class predictions as well. The F1 score of the "1" class however, increased.
- Varying gamma in the rbf kernel lead to an increase of training accuracy on increasing gamma. But, the model started overfitting and after increasing it over 0.01, the test accuracy started to fall. Similarly, on decreasing gamma, the accuracy also decreased.

Accuracies:

- Linear Kernel:
  - Train Accuracy(On a subset of episodes) – 95.2
  - Test Accuracy(On validation dataset) – 92.08

- Gaussian Kernel:
  - Train Accuracy(On a subset of episodes) – 97.14
  - Test Accuracy(On validation dataset) – 90.11

But, for all these cases the F1 score was abysmal (0.06 for linear and 0.00 for rbf)

Part (b) – CNNs:

Libraries Used:

- OpenCV
- PILLOW
- Pickle
- Sklearn
- Keras
- Tensorflow (as Keras backend)

Preprocessing:

- Cropped the image, taking only [33:194, 8:152] to get rid of border and score box
- Also specified the datatype of image array as uint8 to save space

HyperParameter Training:

- Varying the number of kernels and FC units didn't seem to improve the validation set accuracy much, since the slowdown in training and forward prop time made it near impossible to try multiple different architectures. Hence, I tried increasing kernels to 64 and 128, but didn't get much improvement on the Val set (despite significant change on the training set). Similarly, I tried having 2 FC layers, and 2048 units, leading to a similar outcome.

Accuracies:

- Train Accuracy(On a subset of episodes) – 97.2
- Test Accuracy(On validation dataset) – 94.08

For these cases, the F1 score came to around 0.4 on the validation set.

Part 2 – Kaggle Competition:

Model Link

Libraries Used:

- OpenCV
- PILLOW
- Pickle
- Sklearn
- Keras
- Tensorflow (as Keras backend)

Model Architecture:

Model Input – (162*5, 144, 3) (Instead of 15 channels)

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 404, 71, 32)       896
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 202, 35, 32)       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 100, 17, 64)       18496
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 50, 8, 64)         0
_________________________________________________________________
flatten_2 (Flatten)          (None, 25600)             0
_________________________________________________________________
dense_3 (Dense)              (None, 2048)              52430848
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 2049
```

Loss function - binary_cross-entropy
Optimizer - adam

Batch size – 32
Number of Epochs – 3
Train/Val split (from training data) – 90/10
Weighting “1” class 15x more than “0” class while generating training points
Learning Rate – Keras Default