CSC 767 Neural Networks & Deep Learning

Hw2. Image Classification with Convolutional Neural Networks (Keras and/or Tensorflow)

Problem 1: Explore the implementation of CNN with a **color image dataset** by your choice. Datasets as **MNIST, CIFAR-10, CIFAR-100 or MNIST-like and CIFAR-like are excluded as well as datasets included into Adabound_Decay and Example_SMOTE.ipynb files in Help.** As the datasets are usually bigger than you can afford, you may decide use only part of it.

All teams explore different color image datasets. Send me an e-mail to have you dataset approved.

A) Download the original dataset (cut it in advance if necessary). You need not less than 5 - 6 classes with not less than 600 images for each of the classes in the training set, not less than 120 images for each of the classes in the validation set, and not less than 40 images for each of the classes in the test set. These numbers are approximate and you should follow the distribution ratio given below. Split the dataset into **three subsets**: training, validation and a test set following the ratio (70% - 20% -10%). Use dataset analysis and preprocessing as normalization or standardization etc. Represent the classes in **training** and *validation* sets in the following two ways shown below for training set. Make conclusion about the imbalance of the training set and use augmentation to balance it.

Do the same for validation set and testing sets (Figure 1) but do not balance them. Do not include the red frame around the data and the figure.

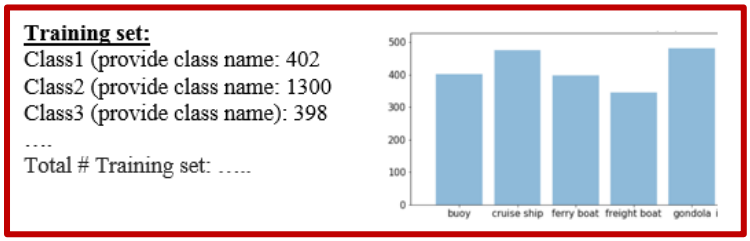


Figure 1. Training set

B) Use Sequential model and build the topology of your **CNN** using

1) Unbalanced training dataset and validation and test datasets.

2) Balanced training set and validation and test datasets.

with CNN having the same topology for both experiments above. Include not less than four convolutional layers, where the number of neurons, the number of filters, pooling layers etc. and is up to you. Include as activation function ReLU, Adam as an optimizer and batch size 32. Do not forget batch normalization and shuffling. Describe each model using comments inside of the code. This will when writing the final report. Apply regularization and dropout to the model to prevent overfitting if necessary.

C) Additionally do 3 more experiments using **ONLY the balanced training dataset** with 3 different models of CNN. The difference could be in the choice of activation function, optimizer, number of convolutional layers, batch size or any combination of these, learning rate change, stride etc.). Include as a first choice ReLU, an optimizer different than Adam and batch size 32. Do not forget batch normalization and shuffling. Describe each model using comments inside of the code. This will when writing the final report. Apply regularization and dropout to the model to prevent overfitting if necessary.

D) Plot training and validation accuracy as well as training and validation loss for each of the above models explained in **Problem 1B** and **Problem 1C. Your graphics should be readable with reasonable size** as shown in Figure 2 and legend should not cross the graphics. **Do not include the red frame around the the figure.**

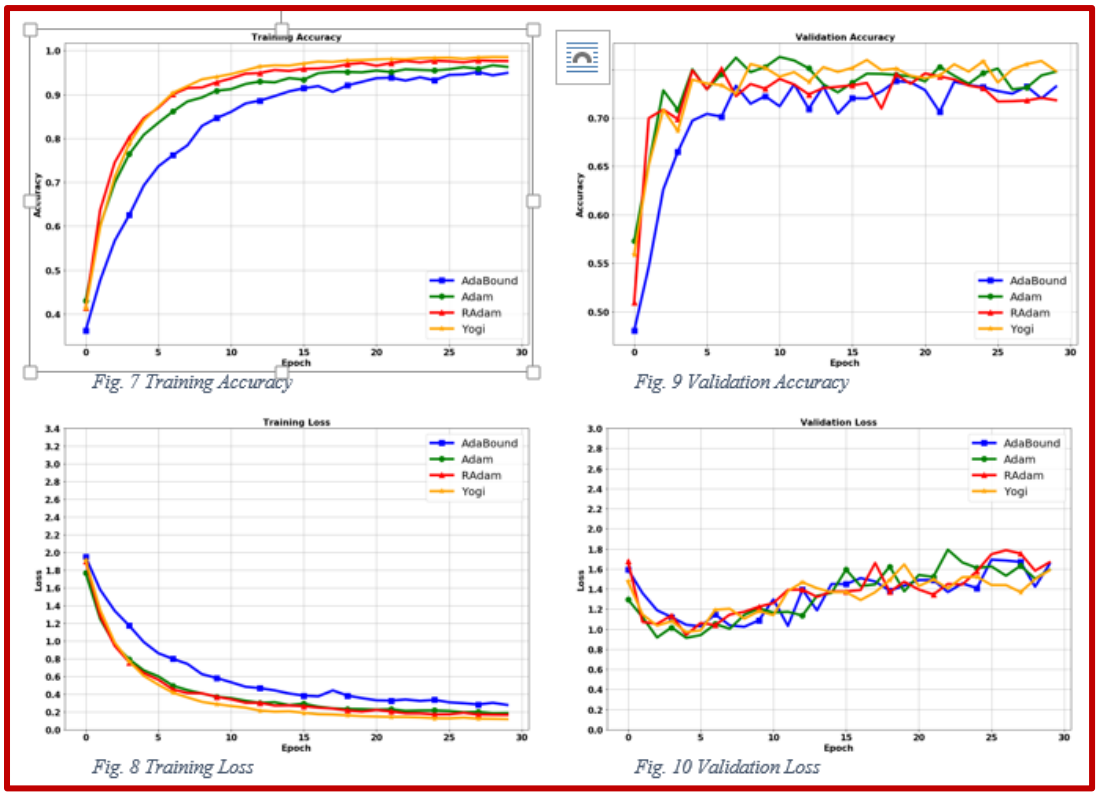


Fig. 7 Training Accuracy *Fig. 9 Validation Accuracy*

Fig. 8 Training Loss *Fig. 10 Validation Loss*

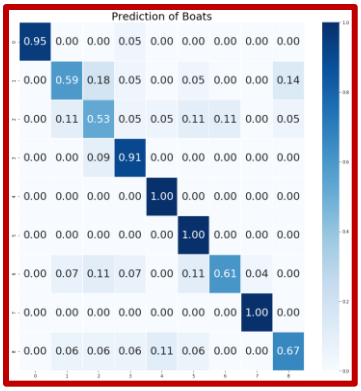
Figure 2. Training and validation accuracy and loss

E) Display some channels in every intermediate activations and explain why this is useful. **(HELP: Deep_Learning_in_Python.pdf).**

F) Visualize (display) convolutional filters: get the gradient of the loss with regard to the input, apply stochastic gradient descent, include a code for filter visualizations and generate a grid of some filter response patterns in a layer. **(HELP: Deep_Learning_in_Python.pdf).**

Grading for full completion of Problem 1: for each part A, B, C, D, E, F – max 20 points
(Max total for Problem 1 = 120 points)

Problem 2. Perform testing. Get the predicted class – show some correctly and not correctly predicted images of testing set as below. **Evaluate test results using NORMALIZED confusion matrix, classification report (precision, recall, f1-score, support, ROC/AUC and Precision-Recall curves). Except tables for this evaluation provide graphics of normalized confusion matrix as shown below, ROC/AUC curves as the one below. (HELP: Adabound_Decay.html). Pick-up the figure sizes and all fonts appropriately (as shown below). Text and numbers must be readable. Do not include the red frame around the figure.**



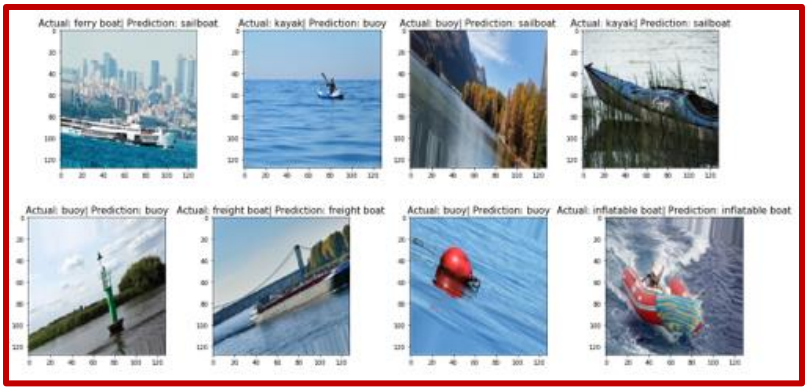
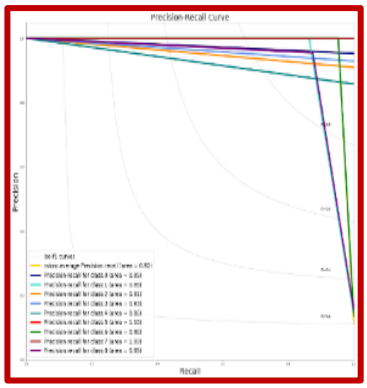


Figure . . . . . **Figure . . . . . .**



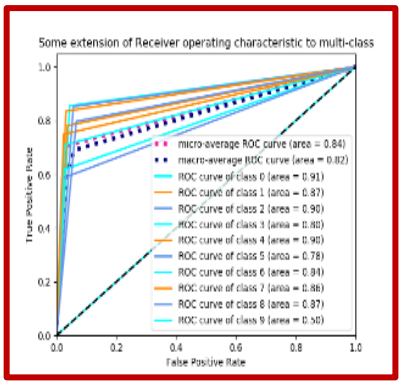


Figure . . . . .

Implement any ENSEMBLING you want and present the results in a table as shown below as well as confusion matrix of your ensemble:

Table 1. . . . . .

Model	Precision	Recall	F-1 Score	AUC
daBound	0.7396	0.7136	0.7200	0.8427
Adam	0.7361	0.7107	0.7158	0.8412
RAdam	0.7470	0.7083	0.7156	0.8402
Yogi	0.7587	0.7420	0.7450	0.8585

Table 2. . . .

Ensemble Model	Precision	Recall	F-1 Score	AUC
Adam, AdaGrad, and Adabound	0.8600	0.8800	0.8600	0.9294
Adam, AdaDelta, and Adabound	0.9000	0.8800	0.8800	0.9329

<u>Grading for full completion of Problem 2</u>: (Max total for Problem 2 = 80 points)

Problem 3: Report: Result Analysis of Problem 1 and Problem 2 (not less than 1-2 full pages with text (pictures and tables are not counted into the two full pages), single space, Times New Roman 11) (max 40 points).

Important Notes:

1. You balance only the training and validation datasets, but not the test set. This is in case you decide to use augmentation in order to remove the imbalance.

2. Confusion matrix must be normalized.

3. Legends included into your graphics should not cross the graphic lines/curves.

4. Pay special attention on the size of your graphics and tables as well on the size of axes values. They should be readable. Look at the Hw2_Description and Help.

Submit on the Blackboard:

a) Upload your dataset or portion of it, all the details like description of the full dataset and link to it (you can use csv or zip files for the complete datasets and doc or pdf file for dataset description).

b) Upload your Python Hw1 Problem1 and Problem 2 files in py, ipynb .pdf/html format (Solution Problem 1&2) which combine your program code and all your outputs.) Including comments into the code will help you in writing your final report.

c) Upload your REPORT (.doc file) which includes the description, analysis and conclusions of Problem 1 and Problem 2 solutions. (Not less than 1-2 full pages with text (pictures and tables are not counted into the two full pages), single space, Times New Roman 11) (max 40 points).

d) List all members of your team and in a separate doc or pdf file shortly explain the contribution of each one. There is no need for each member of the team to upload Hw2 on the Blackboard. But make sure that the submission is done before the expiration of due date and time.

The max number of points for Hw2 is 240 points.

Your e-mail submissions will be ignored. Please, read the Syllabus.