DATA 255 Deep Learning Technologies – Homework -2

Deadline – 11.59 PM – 10/12/2023

20 Points

Problem 1: Apply the following models on the **Fashion Mnist** Dataset. Train the model with the training data and evaluate the model with the test data.

a. **CNN model from scratch:** Develop a CNN model with 5 convolutional layers (with kernel size= 3, stride =1, padding = “same”, activation function = “relu”) with following MaxPooling layer (Size= 2) and 3 fully connected layer (including one output layer). After each of the Convolutional layer apply Batch Normalization. In the fully connected layer apply dropout (rate 0.50). Show the learning curve. Report performance evaluation on the test data. **3 pts**

b. Apply 5-Fold Cross Validation on the CNN model developed in a and report the average accuracy with standard deviation. **2 pts**

c. Apply grid search on the CNN model to find the optimal set of hyperparameters that produce the max performance on the test data. You must train the model using the training data and evaluate model performance using the test dataset. Use grid search for hyperparameter tuning with the following: **3 pts**

Hyperparameters	values
Activation function for Hidden Layer	ReLU, LeakyReLU
optimizer	Adam, Adagrad
Mini-batch size	16,32,64
Learning rate	0.001, 0.0001, 0.00001

You should not use any inbuilt package (like gridsearch) for grid search hyperparameter tuning. Write a for loop and find all possible combinations results. Your output should look like below. Include the output and mention the optimal hyperparameters.

Hyperparameter Combinations (“Activation function for Hidden Layer”, “optimizer”, “mini-batch size”, “learning rate”)	Test Accuracy
ReLU, Adam, 16, 0.001	
…….	

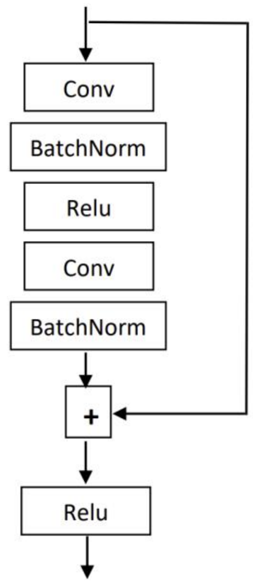
d. **Data Augmentation:** Apply five different image augmentation techniques on the Fashion Mnist train data to augment it and then apply the previously designed (from a) model on it. **2 pts**

e. **Transfer Learning:** Load the VGG-19 model. Drop after the block4 conv1 layer (highlighted in the image below) and on top of it add one **global average pooling** layer, one fully connected layer, and one final output layer. Keep the base model layers (VGG19) freeze. **2 pts**

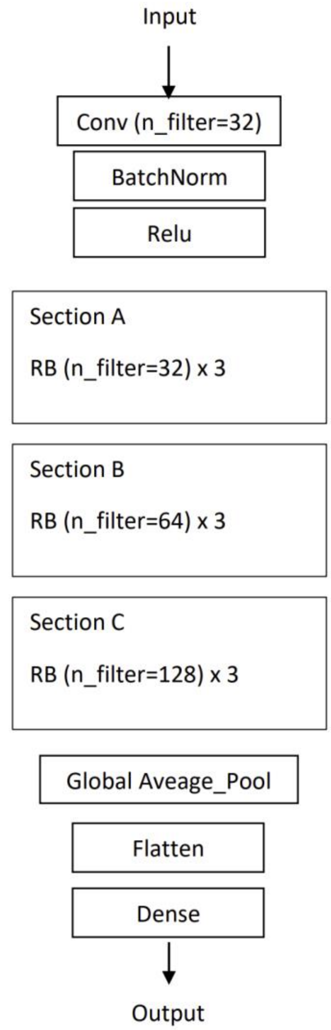
```
Model: "vgg19"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_6 (InputLayer)        [(None, 32, 32, 3)]       0

 block1_conv1 (Conv2D)       (None, 32, 32, 64)        1792

 block1_conv2 (Conv2D)       (None, 32, 32, 64)        36928

 block1_pool (MaxPooling2D)  (None, 16, 16, 64)        0

 block2_conv1 (Conv2D)       (None, 16, 16, 128)       73856

 block2_conv2 (Conv2D)       (None, 16, 16, 128)       147584

 block2_pool (MaxPooling2D)  (None, 8, 8, 128)         0

 block3_conv1 (Conv2D)       (None, 8, 8, 256)         295168

 block3_conv2 (Conv2D)       (None, 8, 8, 256)         590080

 block3_conv3 (Conv2D)       (None, 8, 8, 256)         590080

 block3_conv4 (Conv2D)       (None, 8, 8, 256)         590080

 block3_pool (MaxPooling2D)  (None, 4, 4, 256)         0

 block4_conv1 (Conv2D)       (None, 4, 4, 512)         1180160

 block4_conv2 (Conv2D)       (None, 4, 4, 512)         2359808

 block4_conv3 (Conv2D)       (None, 4, 4, 512)         2359808
```

Problem 2 **(8 pts):** Developing ResNet model from scratch

Apply a residual network specified in the following architecture. All convolutional layers use kernel size 3, stride = 1, and padding = “same”, Residual Block (RB):



ResNet Structure



Save the trained model after training. Your code should not train the model from scratch and return the model. Rather, the code should load a trained model from a model file and return it.

You should not upload the .h5 file of the trained model to Canvas. Instead, you should share it in your google drive and put the share link in a comment line at the beginning of your code. Before submission, make sure people other than yourself can download the model file using the link. Your code should work when your code and the model file are in the same directory.

Useful link-

Data Download:

Fmnist: keras- https://keras.io/api/datasets/fashion_mnist/

CIFAR10: Keras: https://keras.io/api/datasets/cifar10/

Pytorch: https://pytorch.org/vision/stable/generated/torchvision.datasets.CIFAR10.html

Layer concatenation: https://keras.io/api/layers/merging_layers/concatenate/

Model save in keras: https://www.tensorflow.org/guide/keras/save_and_serialize

You are required to submit:

1. An MS/PDF/Scanned document:
    a. Include all the steps of your calculations.
    b. Attach screenshots of the code output.
    c. Include the summary of the model
    d. Include a Table - Mention all the hyperparameters you selected: activation function in hidden layer and output layer, weight initializer, number of hidden layers, neurons in hidden layers, loss function, optimizer, number of epochs, batch size, learning rate, evaluation metric
2. Source code:
    a. Python (Jupyter Notebook)
    b. Ensure it is well-organized with comments and proper indentation.

- Failure to submit the source code will result in a deduction of 5 points.
- Format your filenames as follows: "your_last_name_HW1.pdf" for the document and "your_last_name_HW1_source_code.ipynb" for the source code.
- Before submitting the source code, please double-check that it runs without any errors.
- Must submit the files separately.
- Do not compress into a zip file.
- HW submitted more than 24 hours late will not be accepted for credit.