NEURAL NETWORK & DEEP LEARNING(CS-5720)

(CRN:31196) ASSIGNMENT - 3

Name : Vamsi Krishna Mekala

ID : 700742751 **Date** : 07/25/2023

Github : https://github.com/vamsi-mekala/Neural-networks-assignment-3

Google Drive: https://drive.google.com/file/d/1HFU5CSvk9eC47XUTuDvMj-NBCzCRhE M/view?usp=sharing

Question 1:

Follow the instruction below and then report how the performance changed.(apply all at once)

- Convolutional input layer, 32 feature maps with a size of 3×3 and a rectifier activation function.
- Dropout layer at 20%.
- Convolutional layer, 32 feature maps with a size of 3×3 and a rectifier activation function.
- Max Pool layer with size 2×2.
- Convolutional layer, 64 feature maps with a size of 3×3 and a rectifier activation function.
- Dropout layer at 20%.
- Convolutional layer, 64 feature maps with a size of 3×3 and a rectifier activation function.
- Max Pool layer with size 2×2.
- Convolutional layer, 128 feature maps with a size of 3×3 and a rectifier activation function.
- Dropout layer at 20%.
- Convolutional layer, 128 feature maps with a size of 3×3 and a rectifier activation function.
- Max Pool layer with size 2×2.
- Flatten layer.
- Dropout layer at 20%.
- Fully connected layer with 1024 units and a rectifier activation function.
- Dropout layer at 20%.

- Fully connected layer with 512 units and a rectifier activation function.
- Dropout layer at 20%.
- Fully connected output layer with 10 units and a Softmax activation function

The above specification are coded in as the model shown below:

```
In [6]: model = Sequential()
    model.add(Conv2D(32,(3,3),activation='relu',input_shape=input_shape,padding='same'))
    model.add(Conv2D(32,(3,3),activation='relu',padding='same'))
    model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
    model.add(Conv2D(64,(3,3),activation='relu',padding='same'))
    model.add(Conv2D(64,(3,3),activation='relu',padding='same'))
    model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
    model.add(Conv2D(128,(3,3),activation='relu',padding='same'))
    model.add(Conv2D(128,(3,3),activation='relu',padding='same'))
    model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
    model.add(Dense(1024,activation='relu'))
    model.add(Dense(1024,activation='relu'))
    model.add(Dense(1024,activation='relu'))
    model.add(Dense(num_classes))
    model.add(Activation('softmax'))
```

Then the model is fit to the training data and then the model is evaluated and tested against some test data:

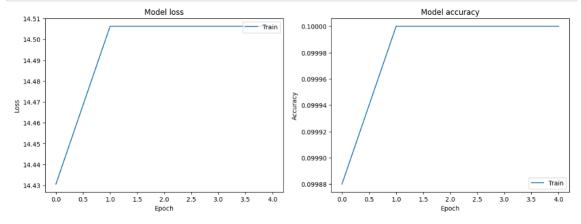
```
In [9]: history = model.fit(x=x_train,y=one_hot_y_train, batch_size=512,epochs=5,verbose=1)
                    WARNING: tensorflow: From c: \users \hp\appdata \local \programs \python \python37 \lib \site-packages \tensorflow \python \
                    y:1250: add_dispatch_support.<locals.wrapper (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a fut
                    ure version.
                    Instructions for updating:
                    Use tf.where in 2.0, which has the same broadcast rule as np.where
                    WARNING:tensorflow:From c:\users\hp\appdata\local\programs\python\python37\lib\site-packages\keras\backend\tensorflow_backend.p
                   y:986: The name tf.assign_add is deprecated. Please use tf.compat.v1.assign_add instead.
                    Epoch 1/5
                    50000/50000 |
                                                         Epoch 2/5
                    50000/50000
                                                                                            =======] - 165s 3ms/step - loss: 14.5063 - acc: 0.1000
                    Epoch 3/5
                    50000/50000 I
                                                                             ========= ] - 175s 3ms/step - loss: 14.5063 - acc: 0.1000
                    Epoch 4/5
```

```
In [19]: import matplotlib.pyplot as plt

# Plot the results
fig, axs = plt.subplots(1, 2, figsize=(15, 5))

# Plot training and validation accuracy
axs[1].plot(history.history['acc'])
axs[1].set_title('Model accuracy')
axs[1].set_ylabel('Accuracy')
axs[1].set_xlabel('Epoch')
axs[1].legend(['Train', 'Test'], loc='lower right')

# Plot training and validation loss
axs[0].plot(history.history['loss'])
axs[0].set_title('Model loss')
axs[0].set_ylabel('Loss')
axs[0].set_ylabel('Loss')
axs[0].set_xlabel('Epoch')
axs[0].legend(['Train', 'Test'], loc='upper right')
plt.show()
```



```
In [21]: # confusion matrix and accuracy
from sklearn.metrics import confusion_matrix, accuracy_score
plt.figure(figsize=(7, 6))
plt.title('Confusion matrix', fontsize=16)
plt.imshow(confusion matrix(y_test, y_predictions1))
classes = ["airplane","automobile","bird","cat","deer","dog","frog","horse","ship","truck"]
plt.xticks(np.arange(10), classes, rotation=45, fontsize=12)
plt.colorbar()
plt.show()
```

