Assignment 13

Referral id: SIRSS1088
NAME: ONASVEE BANARSE
EMAIL : 2obanarse@gmail.com
COLLEGE: AISSMS IOIT
GitHub : https://github.com/ORION-22/RegexSoftware_ASSIGNMENT.git
Kaggle:https://www.kaggle.com/onasveebanarse

In [ ]:
```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import zipfile
import tensorflow as tf
from tensorflow.keras.optimizers import RMSprop
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras import datasets,layers,models,Model
```

In [ ]:
```
# loading data set

(X_train,Y_train),(X_test,Y_test)=datasets.cifar10.load_data()
```

In [ ]:
```
X_train.shape
```

In [ ]:
```
Y_train.shape
```

In [ ]:
```
X_test.shape
```

In [ ]:
```
# reshape  2 d to 1 d
Y_train = Y_train.reshape(-1,)
Y_test = Y_test.reshape(-1,)
```

In [ ]:
```
classes = ["airplane","automobile","bird","cat","deer","dog","frog","horse","ship","truck"]
```

In [ ]:
```
def plot_sample(X, Y, index):
    plt.figure(figsize = (15,2))
    plt.imshow(X[index])
    plt.xlabel(classes[Y[index]])
```

In [ ]:
```
plot_sample(X_train, Y_train, 6)
```

In [ ]:
```
for i in range(10):
  plot_sample(X_train, Y_train, i)
```

In [ ]:
```
# normalization
X_train=X_train/255
X_test=X_test/255
```

Simple ANN

categorical_crossentropy= when y is one hot encoded 0000100 sparse_categorical_crossentropy = when y is a define value y=9

In [ ]:
```
model_ann = models.Sequential([
        layers.Flatten(input_shape=(32,32,3)),
        layers.Dense(3000, activation='relu'),
        layers.Dense(2000, activation='relu'),
        layers.Dense(1000, activation='relu'),
        layers.Dense(10, activation='softmax')
    ])

model_ann.compile(optimizer='SGD',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

MODEL_1=model_ann.fit(
    X_train, Y_train, epochs=10
)
```

ANN=**accuracy: 0.5764**

CNN

In [ ]:
```
model_cnn = models.Sequential([
        layers.Conv2D(filters=32,kernel_size=(3,3),activation='relu',input_shape=(32,32,3)),
        layers.MaxPooling2D((2,2)),

        layers.Conv2D(filters=64,kernel_size=(3,3),activation='relu'),
        layers.MaxPooling2D((2,2)),
        layers.Dropout(0.5),

        layers.Conv2D(filters=128,kernel_size=(3,3),activation='relu'),
        layers.MaxPooling2D((2,2)),
        layers.Dropout(0.5),

        layers.Flatten(),
        layers.Dense(512, activation='relu'),
        layers.Dense(10, activation='softmax')
    ])
```

In [ ]:
```
model_cnn.compile(
                loss = 'sparse_categorical_crossentropy',
               optimizer = RMSprop(learning_rate=1e-4),
               metrics = ['accuracy'])


MODEL_2=model_cnn.fit(X_train,Y_train,
                      epochs=20,
                     verbose=1)
```

In [ ]:
```
model_cnn.evaluate(X_test,Y_test)
```

CNN_RMSprop=

- **accuracy on train: 0.6262**
- **accuracy on test: 0.6644**

In [ ]:
```
model_cnn_2 = models.Sequential([
        layers.Conv2D(filters=32,kernel_size=(3,3),activation='relu',input_shape=(32,32,3)),
        layers.MaxPooling2D((2,2)),

        layers.Conv2D(filters=64,kernel_size=(3,3),activation='relu'),
        layers.MaxPooling2D((2,2)),
        layers.Dropout(0.5),

        layers.Conv2D(filters=128,kernel_size=(3,3),activation='relu'),
        layers.MaxPooling2D((2,2)),
        layers.Dropout(0.5),

        layers.Flatten(),
        layers.Dense(512, activation='relu'),
        layers.Dense(10, activation='softmax')
    ])

model_cnn_2.compile(
                loss = 'sparse_categorical_crossentropy',
               optimizer = "adam",
               metrics = ['accuracy'])


MODEL_3=model_cnn_2.fit(X_train,Y_train,
                      epochs=20,
                     verbose=1)
```

In [ ]:
```
model_cnn_2.evaluate(X_test,Y_test)
```

CNN_adam=

- **accuracy on train: 0.6880**
- **accuracy on test: 0.7342**

In [ ]:
```
y_pred = model_cnn_2.predict(X_test)

y_classes = [np.argmax(element) for element in y_pred]
```

```
y_classes[:5]
```

In [ ]:
```
Y_test[:5]
```

In [ ]:
```
plot_sample(X_test, Y_test,1)
```

In [ ]:
```
preds=[]
for element in y_classes:
    try:
        preds.append(classes[element])
    except:
        preds.append('\0')
ids=[ele+1 for ele in range(len(preds))]
```

In [ ]:
```
submission = pd.DataFrame({'label':preds},index=ids)



submission.to_csv('submission_cifar_10.csv')
```