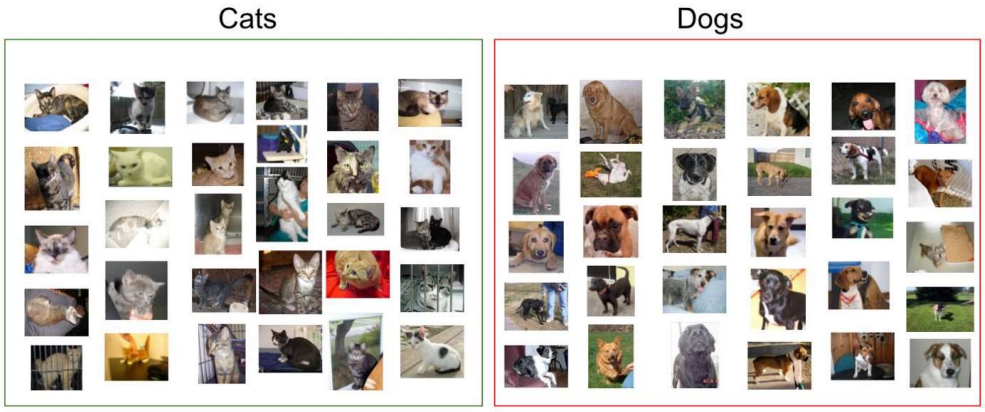
Gabriel Baranes

DEEP LEARNING CS 577

HW 4 : Repport

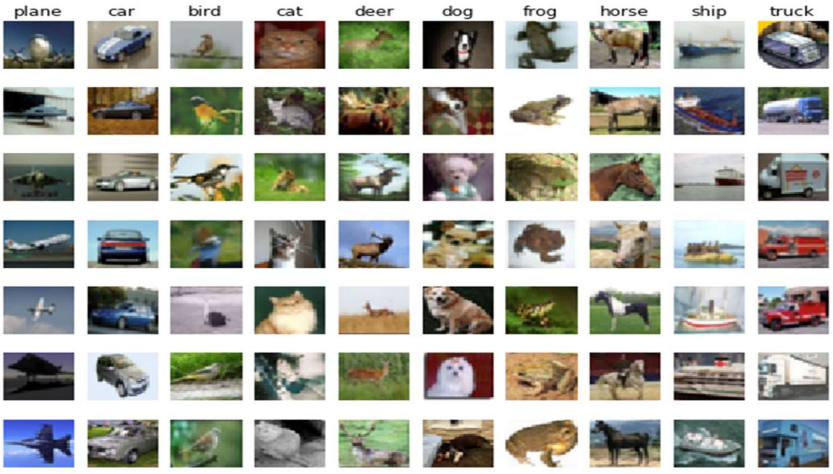
Problem Statement :

For the first question we want to perform Binary classification on the cats and dog image dataset. The goal is to determine weather it is a cat or a dog on the picture.



Sample of cats & dogs images from Kaggle Dataset

The second question is similar to the first one because it also is image classification, but this time it is a multiclass classification. The dataset used is the CIFAR10 dataset.



Solution :

To solve those two classification problems, we will implement and build convolutional neural networks with many convolution, pooling and normalization layers. We will then tune hyper parameters and test our models performances .

Implementation details :

- For the cats and dog dataset:

I first downloaded the dataset and on my computer, I selected the first 2000 images of cats and first 2000 images of dogs and loaded them in my python file, then defined validation, testing and training datasets.

```
x_train = []
x_test = []
y_train = []
y_test = []


#train = []
#test = []
#trainlabel = []
#testlabel = []


#cats
train_image, train_label, test_image, test_label = enr_des_donnees('C:/Users/edbar/Desktop/Illinois Institute of Technology/1 er
x_train.extend(train_image)
y_train.extend(train_label)
x_test.extend(test_image)
y_test.extend(test_label)

#dogs
train_image, train_label, test_image, test_label = enr_des_donnees('C:/Users/edbar/Desktop/Illinois Institute of Technology/1 er
x_train.extend(train_image)
y_train.extend(train_label)
x_test.extend(test_image)
y_test.extend(test_label)

x_train = np.array(x_train)
y_train = np.array(y_train)
x_test = np.array(x_test)
y_test = np.array(y_test)
```

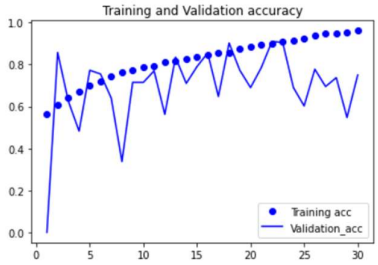
I then reshaped the datas

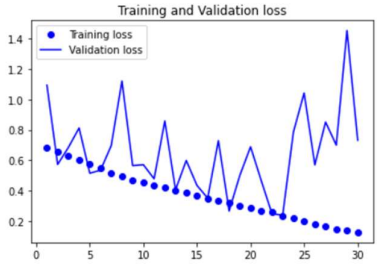
```
x_train = x_train.reshape(x_train.shape[0], 128,128,1).astype('float32')
x_test = x_test.reshape(x_test.shape[0], 128, 128, 1).astype('float32')
x_val=x_val.reshape(x_val.shape[0], 128,128,1).astype('float32')
x_train = x_train/255.0
x_test = x_test/255.0
x_val=x_val/255.0
```

And build the network:

```
model = models.Sequential()
model.add(layers.Conv2D(32, (3,3), activation='relu', input_shape=(128, 128, 1)))
model.add(layers.MaxPooling2D(2,2))
model.add(layers.Conv2D(64, (3,3), activation='relu'))
model.add(layers.MaxPooling2D(2,2))
model.add(layers.Conv2D(128, (3,3), activation='relu'))
model.add(layers.MaxPooling2D(2,2))
model.add(layers.Conv2D(128, (3,3), activation='relu'))
model.add(layers.MaxPooling2D(2,2))
model.add(layers.Flatten())
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
model.summary()
```

I then evaluated and tuned the hyper parameter, and then ploted the graph results.





- For the CIFAR10 Dataset:

I load the data and check their dimensions, the create the validations data

```
(x, y), (x_test, y_test) = datasets.cifar10.load_data()
x_train = x[:48000]
x_val = x[-2000:]

y_train = y[:48000]
y_val = y[-2000:]
```

Dimensions

```
print("x_train",x_train.shape)
print("y_train",y_train.shape)
print("x_val", x_val.shape)
print("y_val", y_val.shape)
print("x_test",x_test.shape)
print("y_test",y_test.shape)
```

```
x_train (48000, 32, 32, 3)
y_train (48000, 1)
x_val (2000, 32, 32, 3)
y_val (2000, 1)
x_test (10000, 32, 32, 3)
y_test (10000, 1)
```

I then build a network with several different layers

```
input = Input(shape=(32, 32, 3))

x = layers.Conv2D(32, (3,3), activation='relu')(input)
x = layers.BatchNormalization()(x)
x = layers.Conv2D(64, (3,3), activation='relu')(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(2)(x)

x = layers.Conv2D(128, (3,3), activation='relu')(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(2)(x)

x = layers.Flatten()(x)

x = layers.Dense(128, activation='relu')(x)
x= layers.Dense(10)(x)
output= activations.softmax(x)

model = Model(inputs=input, outputs=output)
model.summary()
```

And then, I train it and test it using imageDataGenerator previously.

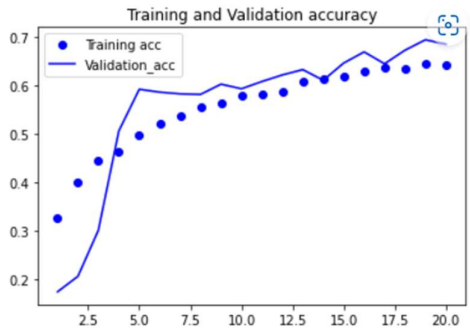
```
donnee_ganerator_train = ImageDataGenerator(featurewise_center=True, featurewise_std_normalization=True, rotation_range=15, widt
donnee_ganerator_train.fit(x_train)
```

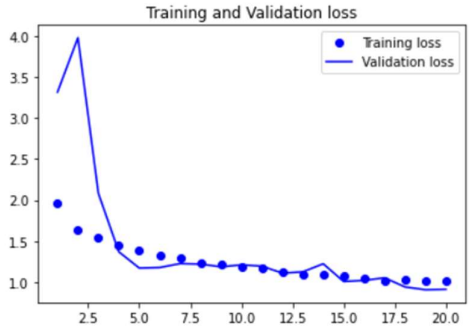
```
donnee_ganerator_test = ImageDataGenerator(featurewise_center=True, featurewise_std_normalization=True)
donnee_ganerator_test.fit(x_val)
donnee_ganerator_test.fit(x_test)
```

convolutional neural network

```
model.compile(loss=losses.SparseCategoricalCrossentropy(),optimizer=optimizers.Adam(),metrics=['accuracy'])
history = model.fit(donnee_ganerator_train.flow(x_train, y_train.flatten(), batch_size=64), steps_per_epoch=len(x_train) / 512,
```

I repeat the training and testing several time to tune the hyper parameter, and once I did that, I plot the results I have with my final model.

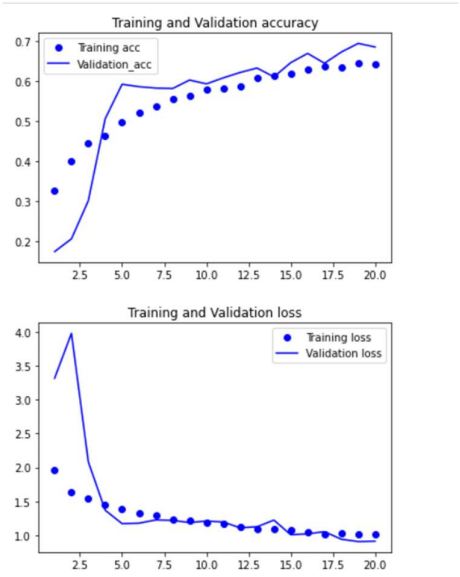




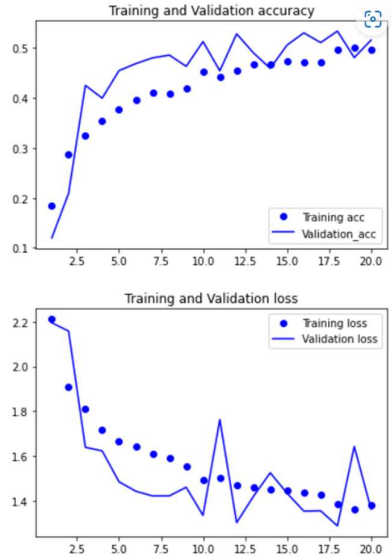
Results and dicussion :

I have had some coding issues and was not able to do all question exactly as asked for the cats and dog data set.

About the CIFAR10 Dataset:



Basic CNN



CNN with inception blocks

The Basic CNN seems to be better than the one with inception blocks because we obtain a better accuracy and smaller loss. But I think my results can be discussed because the training and validation results are very close. Then there might be over fitting.