Basic_Keras_DNN

April 17, 2021

Name: Ventrapragada Sai Shravani
PRN:17070123120
Batch: G-5 (2017-21)

[1]:
```
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense  #Fully connected layer
from keras.layers import Dropout #Neurons get dropped
from keras.layers.normalization import BatchNormalization
from keras.optimizers import SGD
from keras import regularizers
import matplotlib.pyplot as plt
```

Load data

[2]:
```
(x_train,y_train),(x_test,y_test)=mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/mnist.npz
11493376/11490434 [==============================] - 0s 0us/step
```

Preprocess the data

[3]:
```
x_train.shape
```

[3]:
```
(60000, 28, 28)
```

[4]:
```
#Flatting the image using reshape with 32 bit precision
x_train = x_train.reshape(60000, 784).astype('float32')
x_test = x_test.reshape(10000, 784).astype('float32')
```

[5]:
```
# Matrix range is from 0-255
x_train /= 255
x_test /= 255
```

[6]:
```
#One hot encoding
y_train[0]
```

[6]:
```
5
```

[7]:
```
# Every image belonging to one perticular class= 10
n_classes = 10
```

```
y_train = keras.utils.to_categorical(y_train, n_classes)
y_test = keras.utils.to_categorical(y_test, n_classes)
```

[8]:
```
y_train[0]
```

[8]:
```
array([0., 0., 0., 0., 0., 1., 0., 0., 0., 0.], dtype=float32)
```

Build model

[9]:
```
model=Sequential()
```

[10]:
```
model.add(Dense(64,activation='relu',input_shape=(784,)))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(64,activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(10,activation='softmax'))
```

[11]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 64)                50240
_________________________________________________________________
batch_normalization (BatchNo (None, 64)                256
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
batch_normalization_1 (Batch (None, 64)                256
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                650
=================================================================
Total params: 55,562
Trainable params: 55,306
Non-trainable params: 256
_________________________________________________________________
```

Training Parameter can be changed and Non- trainable cant be changed while training

[12]:
```
model.
↪compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

Model training

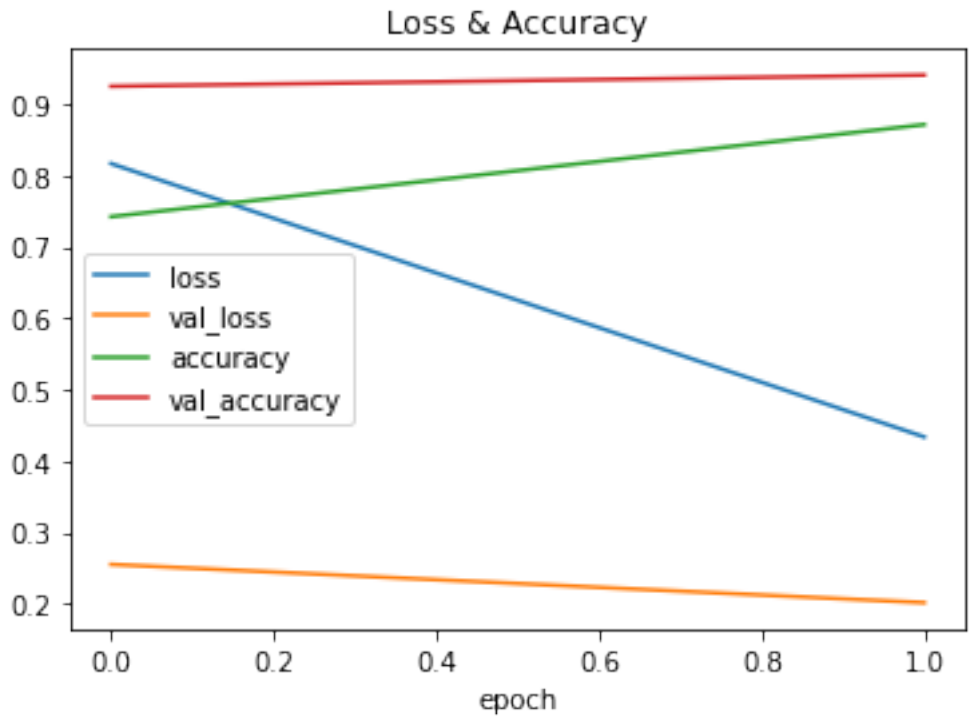
[13]:
```
history= model.fit(x_train, y_train, batch_size=128, epochs=2, verbose=1,␣
↪validation_data=(x_test, y_test))
```

```
Epoch 1/2
469/469 [==============================] - 6s 5ms/step - loss: 1.2667 -
accuracy: 0.5989 - val_loss: 0.2549 - val_accuracy: 0.9246
Epoch 2/2
469/469 [==============================] - 2s 4ms/step - loss: 0.4558 -
accuracy: 0.8656 - val_loss: 0.2013 - val_accuracy: 0.9403
```

Model Evaluation

[15]:
```
import matplotlib.pyplot as plt
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.legend(['loss', 'val_loss', 'accuracy', 'val_accuracy'])
plt.title('Loss & Accuracy')
plt.xlabel('epoch')
```

[15]:
```
Text(0.5, 0, 'epoch')
```



[17]:
```
score = model.evaluate(x_test, y_test, verbose = 1)
print('Test Score', score[0])
print('Test Accuracy', score[1])
```

```
313/313 [==============================] - 1s 2ms/step - loss: 0.2013 -
accuracy: 0.9403
```

```
Test Score 0.20132894814014435
Test Accuracy 0.9402999877929688
```

[ ]:
```
!wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
from colab_pdf import colab_pdf
colab_pdf('Basic_Keras_DNN.ipynb')
```

```
--2021-04-17 07:54:13--  https://raw.githubusercontent.com/brpy/colab-
pdf/master/colab_pdf.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)...
185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com
(raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1864 (1.8K) [text/plain]
Saving to: colab_pdf.py

colab_pdf.py        100%[===================>]   1.82K  --.-KB/s    in 0s

2021-04-17 07:54:14 (29.6 MB/s) - colab_pdf.py saved [1864/1864]

Mounted at /content/drive/

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.


WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Extracting templates from packages: 100%
```