Mnist_CNN

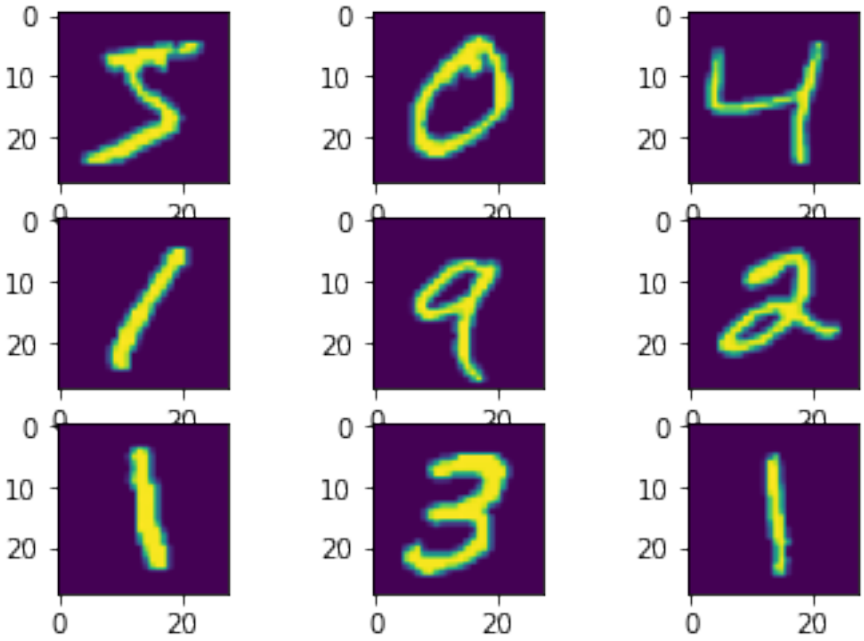
May 18, 2022

[1]:
```
#importing important libraries
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, BatchNormalization, Activation
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.utils import np_utils
from keras.datasets import mnist
import PIL
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential
from tensorflow import keras
```

[3]:
```
(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
11501568/11490434 [==============================] - 0s 0us/step
```

[5]:
```
import matplotlib.pyplot as plt
for i in range(9):
  plt.subplot(330+i+1)
  plt.imshow(x_train[i])
plt.show()
```



[6]:
```
x = x_test
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train = x_train/255
x_test = x_test/255
```

[7]:
```
y_train = np_utils.to_categorical(y_train,10)
y_test = np_utils.to_categorical(y_test,10)
```

[8]:
```
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, BatchNormalization, Activation
from tensorflow.keras.models import Sequential
model=Sequential()
model.
 ↪add(Conv2D(64,(3,3),activation='relu',kernel_initializer='he_uniform',padding='same',input_
model.
 ↪add(Conv2D(64,(3,3),activation='relu',kernel_initializer='he_uniform',padding='same'))
model.add(MaxPooling2D(2,2))
```

[9]:
```
model.add(Flatten())
```

[10]:
```
model.add(Dense(128, activation = 'relu', kernel_initializer =␣
 ↪'he_uniform',input_shape=(28,28)))
model.add(Dropout(0.2))
```

```
model.add(Dense(10,activation='relu'))
model.add(Dropout(0.1))
```

[12]:
```
model.add(Dense(10, activation = 'softmax'))
```

[14]:
```
from tensorflow.keras.optimizers import SGD
opt = SGD(lr = 0.01, momentum = 0.9)
```

```
/usr/local/lib/python3.7/dist-
packages/keras/optimizer_v2/gradient_descent.py:102: UserWarning: The `lr`
argument is deprecated, use `learning_rate` instead.
  super(SGD, self).__init__(name, **kwargs)
```

[15]:
```
from keras.backend import categorical_crossentropy
model.compile(optimizer = opt, loss = 'categorical_crossentropy', metrics =␣
↪['accuracy'])
```

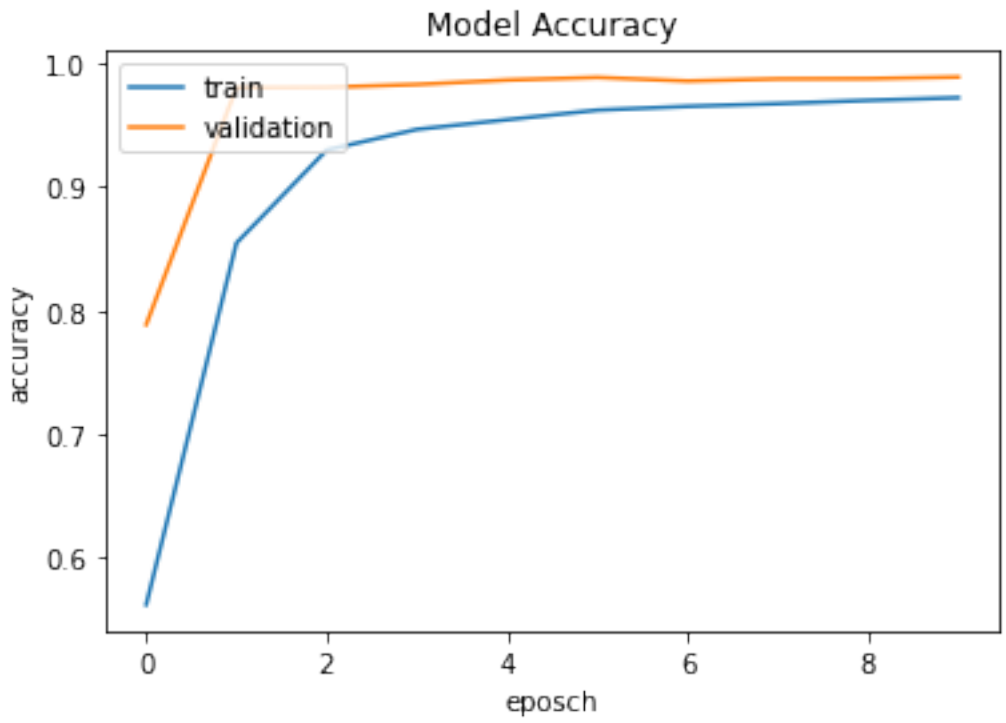
[16]:
```
history = model.fit(x_train, y_train, epochs = 10, batch_size = 128,␣
↪validation_data = (x_test, y_test))
```

```
Epoch 1/10
469/469 [==============================] - 24s 26ms/step - loss: 1.1241 -
accuracy: 0.5620 - val_loss: 0.5854 - val_accuracy: 0.7887
Epoch 2/10
469/469 [==============================] - 10s 22ms/step - loss: 0.4384 -
accuracy: 0.8547 - val_loss: 0.0946 - val_accuracy: 0.9804
Epoch 3/10
469/469 [==============================] - 10s 21ms/step - loss: 0.2497 -
accuracy: 0.9299 - val_loss: 0.0822 - val_accuracy: 0.9807
Epoch 4/10
469/469 [==============================] - 10s 21ms/step - loss: 0.1900 -
accuracy: 0.9467 - val_loss: 0.0686 - val_accuracy: 0.9830
Epoch 5/10
469/469 [==============================] - 10s 21ms/step - loss: 0.1566 -
accuracy: 0.9545 - val_loss: 0.0578 - val_accuracy: 0.9866
Epoch 6/10
469/469 [==============================] - 10s 21ms/step - loss: 0.1350 -
accuracy: 0.9622 - val_loss: 0.0482 - val_accuracy: 0.9887
Epoch 7/10
469/469 [==============================] - 11s 24ms/step - loss: 0.1239 -
accuracy: 0.9653 - val_loss: 0.0641 - val_accuracy: 0.9858
Epoch 8/10
469/469 [==============================] - 10s 21ms/step - loss: 0.1148 -
accuracy: 0.9675 - val_loss: 0.0628 - val_accuracy: 0.9875
Epoch 9/10
469/469 [==============================] - 10s 21ms/step - loss: 0.1060 -
accuracy: 0.9701 - val_loss: 0.0522 - val_accuracy: 0.9877
```

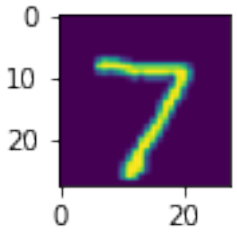
```
Epoch 10/10
469/469 [==============================] - 10s 21ms/step - loss: 0.0960 -
accuracy: 0.9723 - val_loss: 0.0519 - val_accuracy: 0.9890
```

[17]:
```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('accuracy')
plt.xlabel('eposch')
plt.legend(['train','validation'], loc = 'upper left')
```

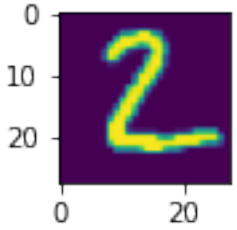
[17]:
```
<matplotlib.legend.Legend at 0x7f4d301ce450>
```



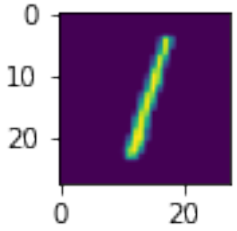
[18]:
```
y_pred=model.predict(x_test)
for i in range(9):
  plt.subplot(330+i+1)
  plt.imshow(x[i])
  plt.show()
  print(np.round(y_pred[i]))
```



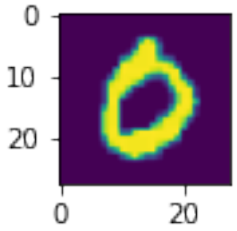

```
[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
```



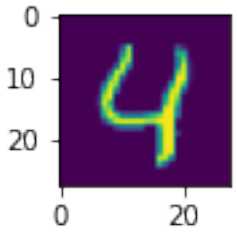

```
[0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
```



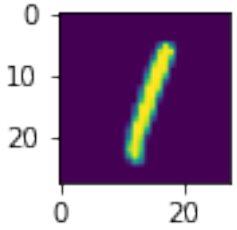

```
[0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
```



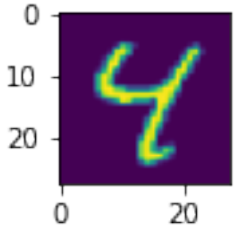

```
[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```



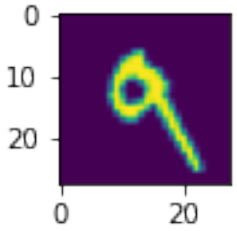

```
[0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
```



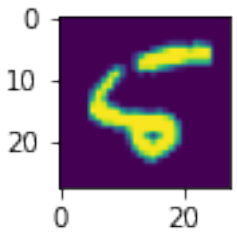

```
[0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
```




```
[0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
```




```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
```




```
[0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
```

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

```
Mounted at /content/drive
--2022-05-18 12:37:24--  https://raw.githubusercontent.com/brpy/colab-
pdf/master/colab_pdf.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)…
185.199.108.133, 185.199.110.133, 185.199.111.133, …
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

2022-05-18 12:37:24 (22.7 MB/s) - ‘colab_pdf.py’ saved [1864/1864]


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%
```

[ ]: