##WE HAVE MNIST DATASET WITH 70000 DATAPOINTS WHERE WE EACH DATA POINT IS 28*28 MATRIX . WHICH IS 784 PIXELS REPRESENTINGNTHE EACH NUMBER RANGING FROM 0 TO 9. IN CASE OF CONVOLUTIONAL NUERAL NETWORKS WE WILL SEND THE IMAGE DATA DIRECTLY AND MODELISE THEM.

In [0]:

```
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense,Dropout,Flatten
from keras.layers import Conv2D,MaxPooling2D
from keras import backend as K
```

```
Using TensorFlow backend.
```

In [0]:

```
batchsize=128
numberofclasses=10
epochs=20
```

In [0]:

```
(xtrain,ytrain),(xtest,ytest)=mnist.load_data()
```

```
Downloading data from https://s3.amazonaws.com/img-datasets/mnist.npz (https://
s3.amazonaws.com/img-datasets/mnist.npz)
11493376/11490434 [==============================] - 1s 0us/step
```

In [0]:

```
print(xtrain.shape)
```

```
(60000, 28, 28)
```

In [0]:

```
img_rows,img_cols=28,28
```

In [0]:

```
if K.image_data_format() == 'channels_first':
    xtrain = xtrain.reshape(xtrain.shape[0], 1, img_rows, img_cols)
    xtest = xtest.reshape(xtest.shape[0], 1, img_rows, img_cols)
    input_shape = (1, img_rows, img_cols)
else:
    xtrain = xtrain.reshape(xtrain.shape[0], img_rows, img_cols, 1)
    xtest = xtest.reshape(xtest.shape[0], img_rows, img_cols, 1)
    input_shape = (img_rows, img_cols, 1)
```

In [0]:

```
print(xtrain.shape)
```

```
(60000, 28, 28, 1)
```

In [0]:

```
xtrain=xtrain.astype('float32')
xtest=xtest.astype('float32')
xtrain/=255
xtest/=255
```

In [0]:

```
from keras.utils import to_categorical
from keras.layers.normalization import BatchNormalization
ytrain=to_categorical(ytrain,10)
ytest=to_categorical(ytest,10)
```

In [0]:

```
import keras as keras
model=Sequential()
```

In [0]:

```
def dynamicplot(x,validationy,testy,ax):
  ax.plot(x,validationy,label='validatiomn loss')
  ax.plot(x,testy,label='testloss')
  plt.xlabel('epoch')
  plt.ylabel('categroicalcrossentropy')
  plt.legend()
  plt.show()
```

In [0]:

```

model.add(Conv2D(100,kernel_size=(3,3),activation='relu',input_shape=input_shape
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(250,(5,5),activation='relu'))
model.add(Dropout(0.25))
model.add(BatchNormalization())
model.add(Flatten())
model.add(Dense(10,activation='softmax'))
model.compile(loss=keras.losses.categorical_crossentropy,optimizer='adam',metric
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pytho
n/framework/op_def_library.py:263: colocate_with (from tensorflow.python.framew
ork.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/te
nsorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) w
ith keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - kee
p_prob`.
```

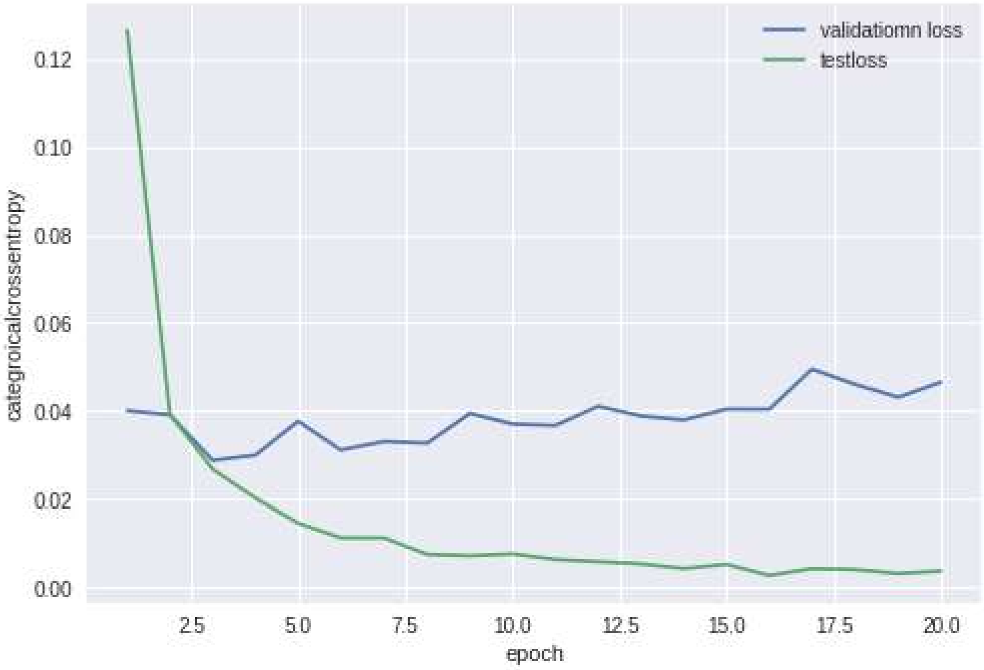
In [0]:

```
history=model.fit(xtrain,ytrain,batch_size=100,epochs=20,verbose=0,validation_da
score=model.evaluate(xtest,ytest,verbose=0)
print('score',score[0])
print('accuaracy',score[1])
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pytho
n/ops/math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is depre
cated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
score 0.046494812891778835
accuaracy 0.9927
```

In [0]:

```
import matplotlib.pyplot as plt
fig,ax=plt.subplots()
x=list(range(1,21))
validationy=history.history['val_loss']
testy=history.history['loss']
dynamicplot(x,validationy,testy,ax)
```



In [0]:

```
import keras as keras
model=Sequential()
model.add(Conv2D(350,kernel_size=(5,5),activation='relu',input_shape=input_shape
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(100,(4,4),activation='relu'))
model.add(Dropout(0.5))
model.add(MaxPooling2D(pool_size=(3,3)))
model.add(Conv2D(75,(3,3),activation='relu'))
model.add(Dropout(0.3))
model.add(Conv2D(40,(1,1),activation='relu'))
model.add(Dropout(0.5))
model.add(Conv2D(25,(1,1),activation='relu'))
model.add(Dropout(0.7))
model.add(Flatten())
model.add(Dense(10,activation='softmax'))
model.compile(loss=keras.losses.categorical_crossentropy,optimizer='adam',metric
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pytho
n/framework/op_def_library.py:263: colocate_with (from tensorflow.python.framew
ork.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/te
nsorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) w
ith keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - kee
p_prob`.
```

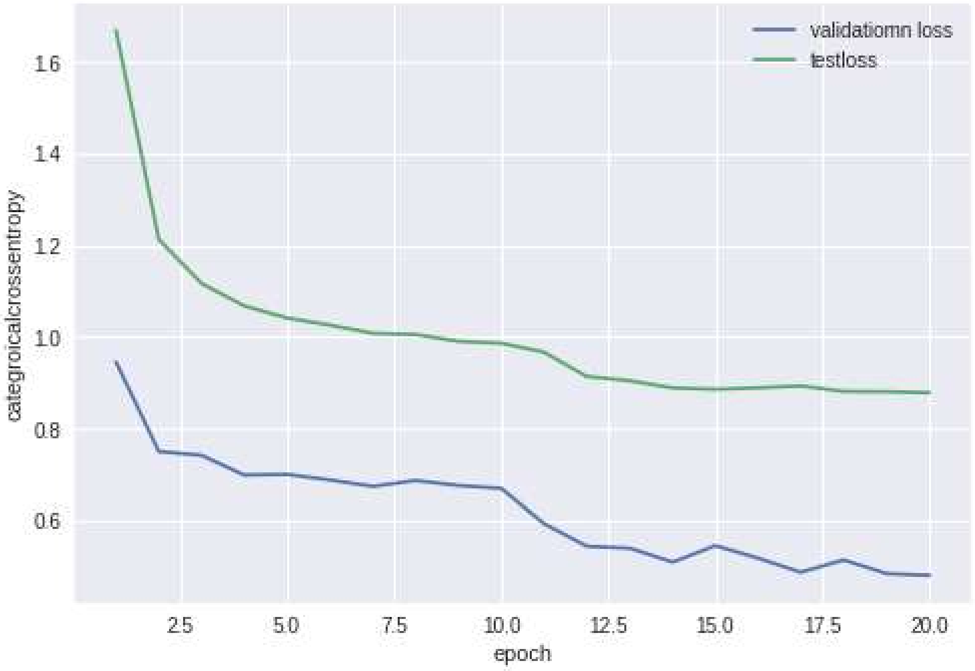
In [0]:

```
history=model.fit(xtrain,ytrain,batch_size=100,epochs=20,verbose=0,validation_da
score=model.evaluate(xtest,ytest,verbose=0)
print('score',score[0])
print('accuaracy',score[1])
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pytho
n/ops/math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is depre
cated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
score 0.4806228316307068
accuaracy 0.8004
```

In [0]:

```
import matplotlib.pyplot as plt
fig,ax=plt.subplots()
x=list(range(1,21))
validationy=history.history['val_loss']
testy=history.history['loss']
dynamicplot(x,validationy,testy,ax)
```



In [0]:

```
import keras as keras
model=Sequential()
model.add(Conv2D(100,kernel_size=(2,2),activation='relu',input_shape=input_shape
model.add(MaxPooling2D(pool_size=(4,4)))
model.add(Conv2D(80,kernel_size=(4,4),activation='relu',input_shape=input_shape)
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))
model.add(BatchNormalization())
model.add(Conv2D(70,kernel_size=(1,1),activation='relu',input_shape=input_shape)
model.add(Conv2D(50,kernel_size=(1,1),activation='relu',input_shape=input_shape)

model.add(Conv2D(60,kernel_size=(1,1),activation='relu',input_shape=input_shape)

model.add(Dropout(0.5))
model.add(Conv2D(100,kernel_size=(1,1),activation='relu',input_shape=input_shape
model.add(Dropout(0.6))
model.add(Conv2D(100,kernel_size=(1,1),activation='relu',input_shape=input_shape
model.add(Conv2D(250,(1,1),activation='relu'))
model.add(Dropout(0.55))
model.add(BatchNormalization())
model.add(Flatten())
model.add(Dense(10,activation='softmax'))
model.compile(loss=keras.losses.categorical_crossentropy,optimizer='adam',metric
```

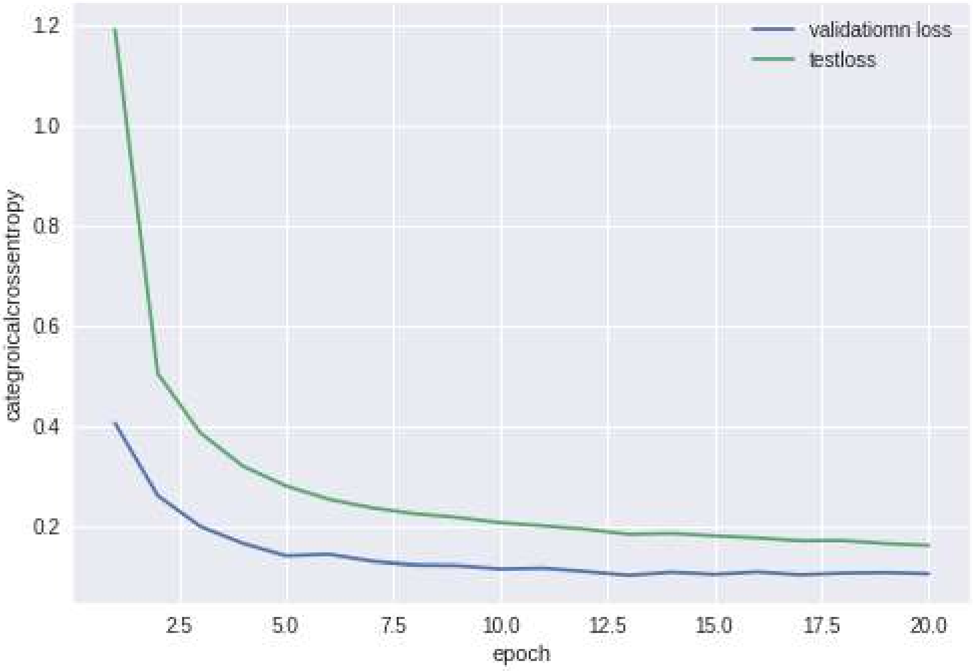
In [0]:

```
history=model.fit(xtrain,ytrain,batch_size=100,epochs=20,verbose=0,validation_da
score=model.evaluate(xtest,ytest,verbose=0)
print('score',score[0])
print('accuaracy',score[1])
```

```
score 0.10515939193330705
accuaracy 0.9751
```

In [0]:

```
fig,ax=plt.subplots()
x=list(range(1,21))
validationy=history.history['val_loss']
testy=history.history['loss']
dynamicplot(x,validationy,testy,ax)
```



In [0]:

```
import pandas as pd

data = [[[100,250],2,0.99,0.04,'yes','yes'],[[350,100,75,40,25],5,0.8,0.48,'yes'
pd.DataFrame(data, columns=["CONV2d layers",'numberof_convolutionallayers', "tes
```

Out[1]:

	CONV2d layers	numberof_convolutionallayers	testaccuracy	test_score	using_dropouts	using_batch_n
1	[100, 250]	2	0.9900	0.04	yes	
2	[350, 100, 75, 40, 25]	5	0.8000	0.48	yes	
3	[100, 80, 70, 50, 60, 100, 100, 250]	8	0.9751	0.10	yes	

CONCLUSIONS,DOCUMENTATION AND KEYTAKEAWAYS

WE NORMALISE THE DATA BEFORE THE DATA SENDING INTO THE CONVOLUTIONAL NUERAL NETWORK AND CONVERSINCE WE HAVE 10 CLASS T THE CLASS LABEL INTO THE CATEGORICAL. IN CONVOLUTIONAL NUERAL NETWORK WE USE FILTERS WHICH ARE USED TO EXTRACT FEATURES FROM THE DATA. WE USE THE MULTIPLE KERNELS WHICH ARE FILTERS . WE USE THE FILTERS TO EXTRACT THE FETURES FROM THE IMAGE THE FEATURES LIKE EDGES ETC....

WE DO THE MAXPOOLING WHICH MEANS REPLACING THE VALUES IN THE MAXIMUM VALUES OF THE SHAPE CONSIDERD.FINALLY WE FLATTEN THE THE NETWORK INTO 10 LAYER SINCE WE HAVE 10 CLASS LABELS.WE USE SOFTMAX AS THE ACTIVATION FUNCTION TO OBTAIN THE PROBABILITIES WE CHOOSE THE VALUE WITH HIGHEST PROBABILITY AS THE OUTPUT.

WE TRY TO MINIMISE THE CATEGORICAL CROSS ENTROPY. WE PLOT THE MODELS FOR TEST AND CROSS VALIDATION LOSSES TO VISUALISE THE HOW THE MODELS ARE CONVERGING REDUCING THE LOSSES WITH THE INCREASED NUMBER OF EPOCHS.

###OBSERVATIONS ARE MODELS WITH MORE NUMBER OF CONVOLUTIONAL NUERAL NETS ARE FASTER CONVERGING THAN THE OTHER MODELS.BUT THE INCREASING THENUMBER OF LAYERS ALWAYS DOES NOT INCREASE ACCURACY THERE MAY THE CHANCES OF OVER FIT IN THE DATA WHICH CAN REDUCE OUR ACCURACY.

WE HAVE ALSO USED THE DROPOUT LAYER AND BATCH NORMALISATION LAYER I WHICH DROPOUT LAYER WE INTODUCE DROPOUT PERCENTAGE BY TURING OF THE CELLS IN THE HIDDEN LAYER TO REDUCE THE CHANCES OF OVER FITTING. WE ALSO USED THE BATCH NORMALSATION LAYER TO NORMALISE BEACIUSE ONCE THE INPUTS ARE SENT INTO NUERAL NETWORK AFTER THE SENDING INTO THE ACTIVATION FUNCTIONS IT MAY LOOSE ITS ORIGINAL BEHAVIOR SO WE NORMLAISE THE DATA . THE MODEL PERFORMANCES ARE DESCRIBED IN THE TABLE ABOVE.

In [0]: