CNN_MNIST_50epoch

August 22, 2018

1 CNN on MINST dataset

1.1 Objective

try 3 different conv 33 55 2*2 kerneals

M1: try 3 conv layer,M2: 5 cond net M3: 7 conv layers

try dropout in some layers

try different maxpool, batch normalization

This is done in google colab

1.2 Import library

```
In [2]: from __future__ import print_function
        import keras
        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
        %matplotlib inline
```

```
Using TensorFlow backend.
```

```
In [3]: batch_size = 128
        num_classes = 10
        epochs = 50

        # the data, split between train and test sets
        (x_train, y_train), (x_test, y_test) = mnist.load_data()

        #x_train=x_train[0:500]
        #y_train=y_train[0:500]
        #x_test=x_test[501:600]
        #y_test=y_test[501:600]
```

```
Downloading data from https://s3.amazonaws.com/img-datasets/mnist.npz
11493376/11490434 [==============================] - 2s 0us/step
```

1.3 Preprocessing

```
In [4]: # input image dimensions
        img_rows, img_cols = 28, 28

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

        x_train = x_train.astype('float32')
        x_test = x_test.astype('float32')
        x_train /= 255
        x_test /= 255
        print('x_train shape:', x_train.shape)
        print(x_train.shape[0], 'train samples')
        print(x_test.shape[0], 'test samples')

        # convert class vectors to binary class matrices
        y_train = keras.utils.to_categorical(y_train, num_classes)
        y_test = keras.utils.to_categorical(y_test, num_classes)

        def plt_dynamic(x, vy, ty, ax, colors=['b']):
            ax.plot(x, vy, 'b', label="Validation Loss")
            ax.plot(x, ty, 'r', label="Train Loss")
            plt.legend()
            plt.grid()
            fig.canvas.draw()
```

```
x_train shape: (60000, 28, 28, 1)
60000 train samples
10000 test samples
```

1.4 3 layer convolution

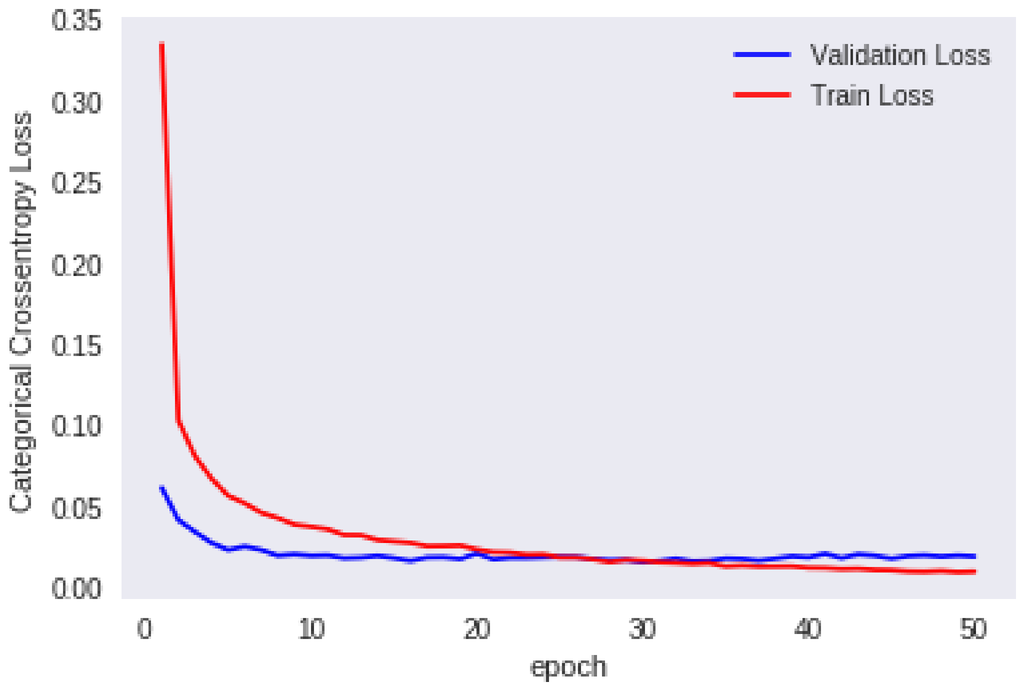
```
In [5]: # Credits: https://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py
        #28*28*1 ===5*5*1 32 conv ===> 24*24*32 ==3*3 64 conv ==> 22*22*64 ==3*3 32 maxpool==>
        #==2*2 64 conv ==> 26*26*64 ==2*2 maxpool ==>25*25 64

        model = Sequential()
        model.add(Conv2D(32, kernel_size=(5, 5),
                         activation='relu',
                         input_shape=input_shape))
        model.add(Conv2D(64, (3, 3), activation='relu'))
        model.add(MaxPooling2D(pool_size=(3, 3)))
        model.add(Conv2D(64, (2, 2), activation='relu'))
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Dropout(0.25))
        model.add(Flatten())#25*25*64
        model.add(Dense(128, activation='relu'))
        model.add(Dropout(0.5))
        model.add(Dense(num_classes, activation='softmax'))
        print(model.summary())

        model.compile(loss=keras.losses.categorical_crossentropy,
                      optimizer=keras.optimizers.Adadelta(),
                      metrics=['accuracy'])

        history=model.fit(x_train, y_train,
                  batch_size=batch_size,
                  epochs=epochs,
                  verbose=1,
                  validation_data=(x_test, y_test))
        score = model.evaluate(x_test, y_test, verbose=0)
        print('Test loss:', score[0])
        print('Test accuracy:', score[1])

        import pandas as pd
        import matplotlib.pyplot as plt
        scoretrain = model.evaluate(x_train, y_train, verbose=0)
        aa=pd.DataFrame()
        bb=pd.DataFrame({'type':['3 conv'],'Test Score':[score[0]],'Test Accuracy':[score[1]],
                           'Train Score':[scoretrain[0]],'Train Accuracy':[scoretrain[1]]})
        aa=aa.append(bb)

        #history = model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs, verbose=1


        fig,ax = plt.subplots(1,1)
        ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')
        x = list(range(1,epochs+1))
        vy = history.history['val_loss']
        ty = history.history['loss']
        plt_dynamic(x, vy, ty, ax)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 24, 24, 32)        832
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 22, 22, 64)        18496
_________________________________________________________________
```

```
max_pooling2d_1 (MaxPooling2 (None, 7, 7, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 6, 6, 64)          16448
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 3, 3, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 3, 3, 64)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 576)               0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               73856
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                1290
=================================================================
Total params: 110,922
Trainable params: 110,922
Non-trainable params: 0
_________________________________________________________________
None
Train on 60000 samples, validate on 10000 samples
Epoch 1/50
60000/60000 [==============================] - 14s 226us/step - loss: 0.3341 - acc: 0.8926 - va
Epoch 2/50
60000/60000 [==============================] - 11s 188us/step - loss: 0.1030 - acc: 0.9696 - va
Epoch 3/50
60000/60000 [==============================] - 11s 187us/step - loss: 0.0810 - acc: 0.9763 - va
Epoch 4/50
60000/60000 [==============================] - 11s 187us/step - loss: 0.0672 - acc: 0.9802 - va
Epoch 5/50
60000/60000 [==============================] - 11s 189us/step - loss: 0.0566 - acc: 0.9829 - va
Epoch 6/50
60000/60000 [==============================] - 11s 190us/step - loss: 0.0519 - acc: 0.9849 - va
Epoch 7/50
60000/60000 [==============================] - 11s 187us/step - loss: 0.0461 - acc: 0.9864 - va
Epoch 8/50
60000/60000 [==============================] - 11s 187us/step - loss: 0.0429 - acc: 0.9870 - va
Epoch 9/50
60000/60000 [==============================] - 11s 187us/step - loss: 0.0389 - acc: 0.9879 - va
Epoch 10/50
60000/60000 [==============================] - 11s 187us/step - loss: 0.0375 - acc: 0.9886 - va
Epoch 11/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0360 - acc: 0.9889 - va
Epoch 12/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0324 - acc: 0.9901 - va
Epoch 13/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0323 - acc: 0.9902 - va
```

```
Epoch 14/50
60000/60000 [==============================] - 11s 184us/step - loss: 0.0293 - acc: 0.9912 - va
Epoch 15/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0285 - acc: 0.9910 - va
Epoch 16/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0276 - acc: 0.9920 - va
Epoch 17/50
60000/60000 [==============================] - 11s 186us/step - loss: 0.0256 - acc: 0.9920 - va
Epoch 18/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0257 - acc: 0.9919 - va
Epoch 19/50
60000/60000 [==============================] - 11s 186us/step - loss: 0.0260 - acc: 0.9921 - va
Epoch 20/50
60000/60000 [==============================] - 11s 186us/step - loss: 0.0235 - acc: 0.9926 - va
Epoch 21/50
60000/60000 [==============================] - 11s 186us/step - loss: 0.0219 - acc: 0.9929 - va
Epoch 22/50
60000/60000 [==============================] - 11s 186us/step - loss: 0.0216 - acc: 0.9931 - va
Epoch 23/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0203 - acc: 0.9933 - va
Epoch 24/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0205 - acc: 0.9934 - va
Epoch 25/50
60000/60000 [==============================] - 11s 186us/step - loss: 0.0186 - acc: 0.9940 - va
Epoch 26/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0186 - acc: 0.9942 - va
Epoch 27/50
60000/60000 [==============================] - 11s 186us/step - loss: 0.0175 - acc: 0.9944 - va
Epoch 28/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0158 - acc: 0.9949 - va
Epoch 29/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0171 - acc: 0.9945 - va
Epoch 30/50
60000/60000 [==============================] - 11s 185us/step - loss: 0.0166 - acc: 0.9948 - va
Epoch 31/50
60000/60000 [==============================] - 11s 184us/step - loss: 0.0157 - acc: 0.9951 - va
Epoch 32/50
60000/60000 [==============================] - 11s 184us/step - loss: 0.0154 - acc: 0.9950 - va
Epoch 33/50
60000/60000 [==============================] - 11s 186us/step - loss: 0.0147 - acc: 0.9957 - va
Epoch 34/50
60000/60000 [==============================] - 11s 187us/step - loss: 0.0152 - acc: 0.9951 - va
Epoch 35/50
60000/60000 [==============================] - 11s 182us/step - loss: 0.0131 - acc: 0.9956 - va
Epoch 36/50
60000/60000 [==============================] - 11s 182us/step - loss: 0.0136 - acc: 0.9954 - va
Epoch 37/50
60000/60000 [==============================] - 11s 181us/step - loss: 0.0130 - acc: 0.9959 - va
```

```
Epoch 38/50
60000/60000 [==============================] - 11s 181us/step - loss: 0.0131 - acc: 0.9959 - va
Epoch 39/50
60000/60000 [==============================] - 11s 181us/step - loss: 0.0132 - acc: 0.9954 - va
Epoch 40/50
60000/60000 [==============================] - 11s 182us/step - loss: 0.0122 - acc: 0.9961 - va
Epoch 41/50
60000/60000 [==============================] - 11s 182us/step - loss: 0.0121 - acc: 0.9960 - va
Epoch 42/50
60000/60000 [==============================] - 11s 182us/step - loss: 0.0115 - acc: 0.9963 - va
Epoch 43/50
60000/60000 [==============================] - 11s 181us/step - loss: 0.0117 - acc: 0.9959 - va
Epoch 44/50
60000/60000 [==============================] - 11s 181us/step - loss: 0.0110 - acc: 0.9962 - va
Epoch 45/50
60000/60000 [==============================] - 11s 182us/step - loss: 0.0108 - acc: 0.9967 - va
Epoch 46/50
60000/60000 [==============================] - 11s 182us/step - loss: 0.0100 - acc: 0.9966 - va
Epoch 47/50
60000/60000 [==============================] - 11s 183us/step - loss: 0.0098 - acc: 0.9970 - va
Epoch 48/50
60000/60000 [==============================] - 11s 181us/step - loss: 0.0103 - acc: 0.9966 - va
Epoch 49/50
60000/60000 [==============================] - 11s 181us/step - loss: 0.0096 - acc: 0.9969 - va
Epoch 50/50
60000/60000 [==============================] - 11s 182us/step - loss: 0.0099 - acc: 0.9967 - va
Test loss: 0.01919088124151722
Test accuracy: 0.9949
```



1.5 5 Layer Convolution

```
In [6]: # Credits: https://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py
        #28*28*1 ===5*5*1 32 conv ===> 24*24*32 ==5*5 64 conv ==> 20*20*64 ==3*3 32 maxpool==>
        #=4*4  128 conv ==> 15*15 128 ==3*3 64 conv ==> 13*13 64 ==2*2 64 maxpool ==>11*11 64
        #= 3*3 64 conv ==>9*9 64 ==2*2 64==>8*8 64

        model = Sequential()
        model.add(Conv2D(32, kernel_size=(5, 5),activation='relu',input_shape=input_shape))
        model.add(Conv2D(64, (5, 5), activation='relu'))
        model.add(MaxPooling2D(pool_size=(3, 3),strides=1))
        model.add(Conv2D(128, (4, 4), activation='relu'))
        model.add(Conv2D(64, (3, 3), activation='relu'))
        #odel.add(ZeroPadding2D((1, 1), input_shape=(img_rows, img_cols, channel)))
        model.add(MaxPooling2D(pool_size=(2, 2),dim_ordering="tf"))

        model.add(Conv2D(64, (3, 3), activation='relu'))
        model.add(MaxPooling2D(pool_size=(4,4)))

        model.add(Dropout(0.25))
        model.add(Flatten())#25*25*64
        model.add(Dense(128, activation='relu'))
        model.add(Dropout(0.5))
        model.add(Dense(num_classes, activation='softmax'))
```

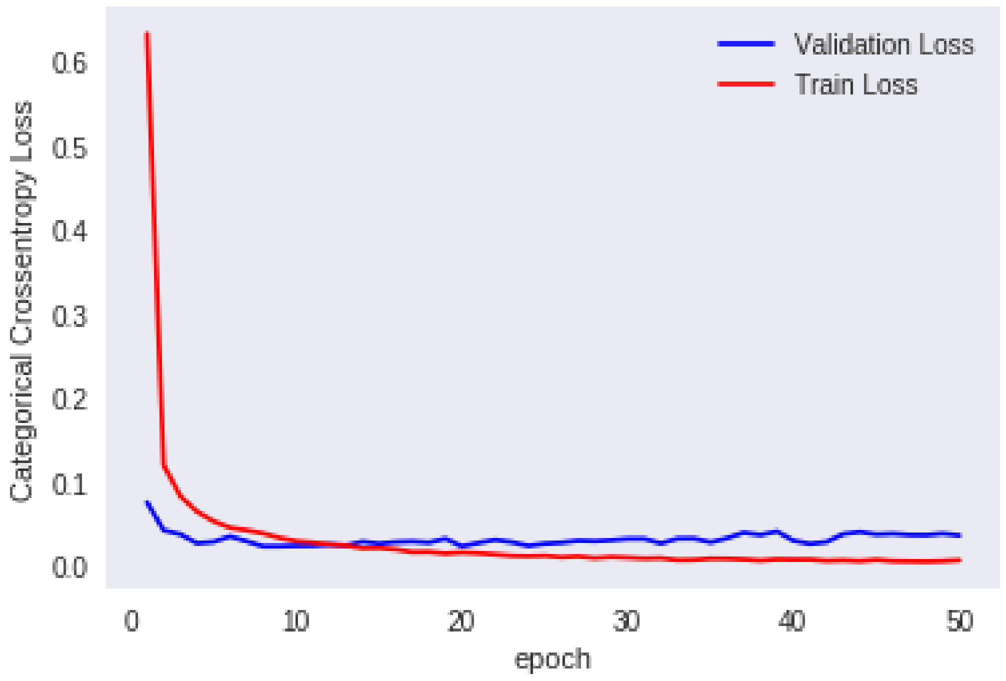
```
print(model.summary())



model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])

history=model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(x_test, y_test))
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])

import pandas as pd
import matplotlib.pyplot as plt
scoretrain = model.evaluate(x_train, y_train, verbose=0)
bb=pd.DataFrame({'type':['5 conv'],'Test Score':[score[0]],'Test Accuracy':[score[1]],
                   'Train Score':[scoretrain[0]],'Train Accuracy':[scoretrain[1]]})
aa=aa.append(bb)

#history = model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs, verbose=1


fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')
x = list(range(1,epochs+1))
vy = history.history['val_loss']
ty = history.history['loss']
plt_dynamic(x, vy, ty, ax)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:9: UserWarning: Update your `MaxP
  if __name__ == '__main__':


_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_4 (Conv2D)            (None, 24, 24, 32)        832       
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 20, 20, 64)        51264     
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 18, 18, 64)        0         
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 15, 15, 128)       131200    
```

```
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 13, 13, 64)        73792
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 4, 4, 64)          36928
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 1, 1, 64)          0
_________________________________________________________________
dropout_3 (Dropout)          (None, 1, 1, 64)          0
_________________________________________________________________
flatten_2 (Flatten)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 128)               8320
_________________________________________________________________
dropout_4 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_4 (Dense)              (None, 10)                1290
=================================================================
Total params: 303,626
Trainable params: 303,626
Non-trainable params: 0
_________________________________________________________________
None
Train on 60000 samples, validate on 10000 samples
Epoch 1/50
60000/60000 [==============================] - 22s 367us/step - loss: 0.6315 - acc: 0.7903 - va
Epoch 2/50
60000/60000 [==============================] - 21s 348us/step - loss: 0.1194 - acc: 0.9676 - va
Epoch 3/50
60000/60000 [==============================] - 21s 348us/step - loss: 0.0828 - acc: 0.9778 - va
Epoch 4/50
60000/60000 [==============================] - 21s 347us/step - loss: 0.0645 - acc: 0.9826 - va
Epoch 5/50
60000/60000 [==============================] - 21s 348us/step - loss: 0.0530 - acc: 0.9859 - va
Epoch 6/50
60000/60000 [==============================] - 21s 352us/step - loss: 0.0453 - acc: 0.9876 - va
Epoch 7/50
60000/60000 [==============================] - 21s 348us/step - loss: 0.0422 - acc: 0.9888 - va
Epoch 8/50
60000/60000 [==============================] - 21s 347us/step - loss: 0.0386 - acc: 0.9899 - va
Epoch 9/50
60000/60000 [==============================] - 21s 347us/step - loss: 0.0332 - acc: 0.9909 - va
Epoch 10/50
60000/60000 [==============================] - 21s 348us/step - loss: 0.0292 - acc: 0.9919 - va
Epoch 11/50
60000/60000 [==============================] - 21s 348us/step - loss: 0.0276 - acc: 0.9926 - va
Epoch 12/50
```

```
60000/60000 [==============================] - 21s 347us/step - loss: 0.0254 - acc: 0.9929 - va
Epoch 13/50
60000/60000 [==============================] - 21s 348us/step - loss: 0.0240 - acc: 0.9936 - va
Epoch 14/50
60000/60000 [==============================] - 21s 348us/step - loss: 0.0209 - acc: 0.9942 - va
Epoch 15/50
60000/60000 [==============================] - 21s 348us/step - loss: 0.0212 - acc: 0.9945 - va
Epoch 16/50
60000/60000 [==============================] - 21s 347us/step - loss: 0.0195 - acc: 0.9946 - va
Epoch 17/50
60000/60000 [==============================] - 21s 347us/step - loss: 0.0164 - acc: 0.9954 - va
Epoch 18/50
60000/60000 [==============================] - 21s 347us/step - loss: 0.0166 - acc: 0.9955 - va
Epoch 19/50
60000/60000 [==============================] - 21s 347us/step - loss: 0.0146 - acc: 0.9957 - va
Epoch 20/50
60000/60000 [==============================] - 21s 347us/step - loss: 0.0159 - acc: 0.9957 - va
Epoch 21/50
60000/60000 [==============================] - 21s 348us/step - loss: 0.0149 - acc: 0.9960 - va
Epoch 22/50
60000/60000 [==============================] - 21s 345us/step - loss: 0.0135 - acc: 0.9965 - va
Epoch 23/50
60000/60000 [==============================] - 21s 346us/step - loss: 0.0119 - acc: 0.9968 - va
Epoch 24/50
60000/60000 [==============================] - 21s 347us/step - loss: 0.0115 - acc: 0.9968 - va
Epoch 25/50
60000/60000 [==============================] - 21s 346us/step - loss: 0.0120 - acc: 0.9965 - va
Epoch 26/50
60000/60000 [==============================] - 21s 345us/step - loss: 0.0102 - acc: 0.9970 - va
Epoch 27/50
60000/60000 [==============================] - 21s 347us/step - loss: 0.0113 - acc: 0.9969 - va
Epoch 28/50
60000/60000 [==============================] - 21s 345us/step - loss: 0.0092 - acc: 0.9975 - va
Epoch 29/50
60000/60000 [==============================] - 21s 345us/step - loss: 0.0106 - acc: 0.9970 - va
Epoch 30/50
60000/60000 [==============================] - 21s 345us/step - loss: 0.0097 - acc: 0.9975 - va
Epoch 31/50
60000/60000 [==============================] - 21s 345us/step - loss: 0.0087 - acc: 0.9975 - va
Epoch 32/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0092 - acc: 0.9974 - va
Epoch 33/50
60000/60000 [==============================] - 21s 345us/step - loss: 0.0069 - acc: 0.9982 - va
Epoch 34/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0072 - acc: 0.9980 - va
Epoch 35/50
60000/60000 [==============================] - 21s 345us/step - loss: 0.0086 - acc: 0.9977 - va
Epoch 36/50
```

```
60000/60000 [==============================] - 21s 346us/step - loss: 0.0084 - acc: 0.9978 - va
Epoch 37/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0077 - acc: 0.9980 - va
Epoch 38/50
60000/60000 [==============================] - 21s 345us/step - loss: 0.0064 - acc: 0.9983 - va
Epoch 39/50
60000/60000 [==============================] - 21s 343us/step - loss: 0.0078 - acc: 0.9980 - va
Epoch 40/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0075 - acc: 0.9981 - va
Epoch 41/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0077 - acc: 0.9980 - va
Epoch 42/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0060 - acc: 0.9984 - va
Epoch 43/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0066 - acc: 0.9983 - va
Epoch 44/50
60000/60000 [==============================] - 21s 343us/step - loss: 0.0056 - acc: 0.9986 - va
Epoch 45/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0073 - acc: 0.9981 - va
Epoch 46/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0055 - acc: 0.9987 - va
Epoch 47/50
60000/60000 [==============================] - 21s 343us/step - loss: 0.0052 - acc: 0.9986 - va
Epoch 48/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0050 - acc: 0.9987 - va
Epoch 49/50
60000/60000 [==============================] - 21s 343us/step - loss: 0.0055 - acc: 0.9985 - va
Epoch 50/50
60000/60000 [==============================] - 21s 344us/step - loss: 0.0067 - acc: 0.9983 - va
Test loss: 0.03613459114649261
Test accuracy: 0.9946
```



2 7Layer Conv

```
In [7]: # Credits: https://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py

        model = Sequential()
        model.add(Conv2D(128, kernel_size=(4, 4),strides=1,activation='relu',input_shape=input_
        model.add(Conv2D(64, (3, 3), activation='relu'))
        model.add(Conv2D(32, (3, 3), activation='relu'))
        model.add(Conv2D(32, (3, 3), activation='relu'))
        model.add(MaxPooling2D(pool_size=(2, 2),strides=1))
        model.add(Conv2D(128, (4, 4), activation='relu'))
        model.add(Conv2D(64, (3, 3), activation='relu'))
        model.add(Conv2D(64, (2, 3), activation='relu'))
        #odel.add(ZeroPadding2D((1, 1), input_shape=(img_rows, img_cols, channel)))
        model.add(MaxPooling2D(pool_size=(2, 2),dim_ordering="tf"))

        model.add(Dropout(0.25))
        model.add(Flatten())#25*25*64
        model.add(Dense(128, activation='relu'))
        model.add(Dropout(0.5))
        model.add(Dense(num_classes, activation='softmax'))
        print(model.summary())
```

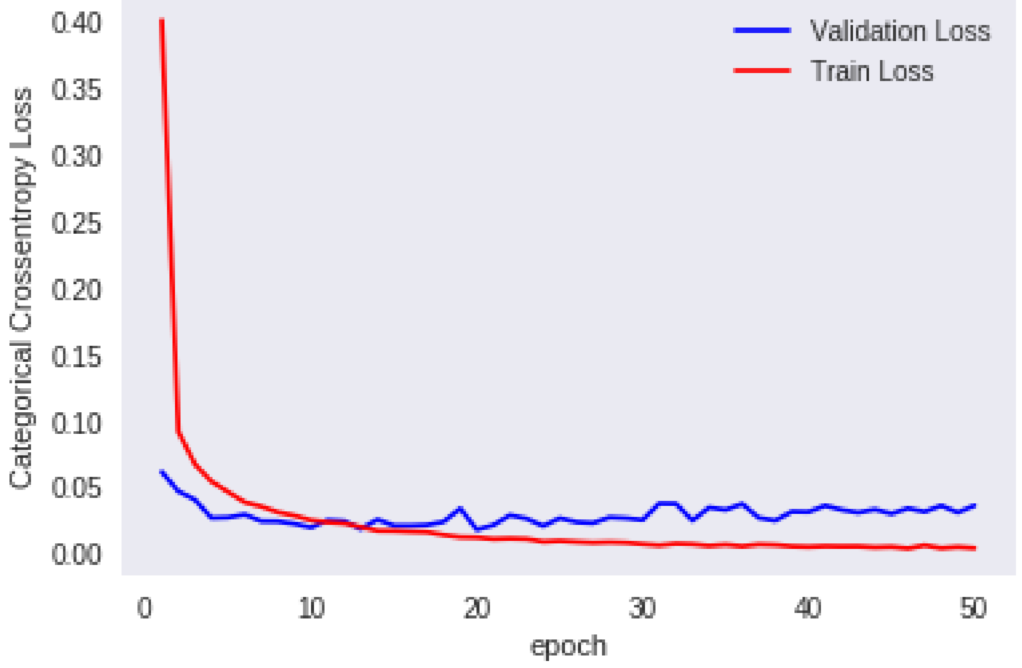
```
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])

history=model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(x_test, y_test))
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])

import pandas as pd
import matplotlib.pyplot as plt
scoretrain = model.evaluate(x_train, y_train, verbose=0)
bb=pd.DataFrame({'type':['7 conv'],'Test Score':[score[0]],'Test Accuracy':[score[1]],
                   'Train Score':[scoretrain[0]],'Train Accuracy':[scoretrain[1]]})
aa=aa.append(bb)

#history = model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs, verbose=1


fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')
x = list(range(1,epochs+1))
vy = history.history['val_loss']
ty = history.history['loss']
plt_dynamic(x, vy, ty, ax)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:12: UserWarning: Update your `MaxI
  if sys.path[0] == '':



_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_9 (Conv2D)            (None, 25, 25, 128)       2176
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 23, 23, 64)        73792
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 21, 21, 32)        18464
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 19, 19, 32)        9248
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 18, 18, 32)        0
_________________________________________________________________
```

```
conv2d_13 (Conv2D)           (None, 15, 15, 128)       65664
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 13, 13, 64)        73792
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 12, 11, 64)        24640
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 6, 5, 64)          0
_________________________________________________________________
dropout_5 (Dropout)          (None, 6, 5, 64)          0
_________________________________________________________________
flatten_3 (Flatten)          (None, 1920)              0
_________________________________________________________________
dense_5 (Dense)              (None, 128)               245888
_________________________________________________________________
dropout_6 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_6 (Dense)              (None, 10)                1290
=================================================================
Total params: 514,954
Trainable params: 514,954
Non-trainable params: 0
_________________________________________________________________
None
Train on 60000 samples, validate on 10000 samples
Epoch 1/50
60000/60000 [==============================] - 33s 545us/step - loss: 0.4001 - acc: 0.8739 - va
Epoch 2/50
60000/60000 [==============================] - 31s 520us/step - loss: 0.0911 - acc: 0.9733 - va
Epoch 3/50
60000/60000 [==============================] - 31s 521us/step - loss: 0.0666 - acc: 0.9804 - va
Epoch 4/50
60000/60000 [==============================] - 31s 520us/step - loss: 0.0537 - acc: 0.9840 - va
Epoch 5/50
60000/60000 [==============================] - 31s 520us/step - loss: 0.0458 - acc: 0.9869 - va
Epoch 6/50
60000/60000 [==============================] - 31s 520us/step - loss: 0.0380 - acc: 0.9884 - va
Epoch 7/50
60000/60000 [==============================] - 31s 519us/step - loss: 0.0349 - acc: 0.9893 - va
Epoch 8/50
60000/60000 [==============================] - 31s 519us/step - loss: 0.0306 - acc: 0.9910 - va
Epoch 9/50
60000/60000 [==============================] - 31s 520us/step - loss: 0.0281 - acc: 0.9913 - va
Epoch 10/50
60000/60000 [==============================] - 31s 522us/step - loss: 0.0245 - acc: 0.9929 - va
Epoch 11/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0231 - acc: 0.9929 - va
Epoch 12/50
60000/60000 [==============================] - 31s 519us/step - loss: 0.0222 - acc: 0.9934 - va
```

```
Epoch 13/50
60000/60000 [==============================] - 31s 519us/step - loss: 0.0195 - acc: 0.9941 - va
Epoch 14/50
60000/60000 [==============================] - 31s 519us/step - loss: 0.0166 - acc: 0.9948 - va
Epoch 15/50
60000/60000 [==============================] - 31s 518us/step - loss: 0.0165 - acc: 0.9953 - va
Epoch 16/50
60000/60000 [==============================] - 31s 518us/step - loss: 0.0160 - acc: 0.9950 - va
Epoch 17/50
60000/60000 [==============================] - 31s 519us/step - loss: 0.0156 - acc: 0.9954 - va
Epoch 18/50
60000/60000 [==============================] - 31s 520us/step - loss: 0.0135 - acc: 0.9960 - va
Epoch 19/50
60000/60000 [==============================] - 31s 519us/step - loss: 0.0119 - acc: 0.9965 - va
Epoch 20/50
60000/60000 [==============================] - 31s 523us/step - loss: 0.0116 - acc: 0.9965 - va
Epoch 21/50
60000/60000 [==============================] - 31s 519us/step - loss: 0.0104 - acc: 0.9969 - va
Epoch 22/50
60000/60000 [==============================] - 31s 518us/step - loss: 0.0108 - acc: 0.9967 - va
Epoch 23/50
60000/60000 [==============================] - 31s 520us/step - loss: 0.0104 - acc: 0.9968 - va
Epoch 24/50
60000/60000 [==============================] - 31s 520us/step - loss: 0.0084 - acc: 0.9973 - va
Epoch 25/50
60000/60000 [==============================] - 31s 519us/step - loss: 0.0090 - acc: 0.9973 - va
Epoch 26/50
60000/60000 [==============================] - 31s 518us/step - loss: 0.0083 - acc: 0.9975 - va
Epoch 27/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0077 - acc: 0.9976 - va
Epoch 28/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0080 - acc: 0.9976 - va
Epoch 29/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0077 - acc: 0.9978 - va
Epoch 30/50
60000/60000 [==============================] - 31s 521us/step - loss: 0.0066 - acc: 0.9980 - va
Epoch 31/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0056 - acc: 0.9982 - va
Epoch 32/50
60000/60000 [==============================] - 31s 516us/step - loss: 0.0070 - acc: 0.9978 - va
Epoch 33/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0065 - acc: 0.9980 - va
Epoch 34/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0052 - acc: 0.9984 - va
Epoch 35/50
60000/60000 [==============================] - 31s 518us/step - loss: 0.0062 - acc: 0.9983 - va
Epoch 36/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0049 - acc: 0.9987 - va
```

```
Epoch 37/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0063 - acc: 0.9983 - va
Epoch 38/50
60000/60000 [==============================] - 31s 519us/step - loss: 0.0059 - acc: 0.9981 - va
Epoch 39/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0049 - acc: 0.9985 - va
Epoch 40/50
60000/60000 [==============================] - 31s 521us/step - loss: 0.0045 - acc: 0.9987 - va
Epoch 41/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0050 - acc: 0.9984 - va
Epoch 42/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0048 - acc: 0.9987 - va
Epoch 43/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0049 - acc: 0.9984 - va
Epoch 44/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0042 - acc: 0.9986 - va
Epoch 45/50
60000/60000 [==============================] - 31s 516us/step - loss: 0.0046 - acc: 0.9987 - va
Epoch 46/50
60000/60000 [==============================] - 31s 518us/step - loss: 0.0033 - acc: 0.9987 - va
Epoch 47/50
60000/60000 [==============================] - 31s 517us/step - loss: 0.0057 - acc: 0.9983 - va
Epoch 48/50
60000/60000 [==============================] - 31s 518us/step - loss: 0.0035 - acc: 0.9990 - va
Epoch 49/50
60000/60000 [==============================] - 31s 518us/step - loss: 0.0044 - acc: 0.9988 - va
Epoch 50/50
60000/60000 [==============================] - 31s 522us/step - loss: 0.0035 - acc: 0.9989 - va
Test loss: 0.03556783508596966
Test accuracy: 0.9949
```



```
In [0]: # Compare
```

2.1 Compare different model

It seems for 3 layer conv net 30 epoch is best, for 5 layer cond net 12 epoch are best and for 5 layer conv net 12 epoch are best, because there cv+train error are equal. Below are the differnent matric for models

```
In [9]: aa
```

```
Out[9]:    Test Accuracy  Test Score  Train Accuracy  Train Score    type
        0         0.9949    0.019191        0.999667     0.001329  3 conv
        0         0.9946    0.036135        0.999533     0.001360  5 conv
        0         0.9949    0.035568        0.999817     0.000580  7 conv
```