In [1]:
```
from __future__ import print_function
import sys
sys.path.append('D:\\Dropbox\\Dropbox\\Master Theoretische Informatik\\Very Deep Learning\\WiSe 18_19\\VDL_Python\Blatt 2\\Very-Deep-Learning-CNN\\')
sys.path.append('I:\\Programme\\Python 3.5\\Lib\\site-packages\\')
```

In [2]:
```
%matplotlib inline
%reload_ext autoreload
```

In [3]:
```
import torch
import torch.nn as nn
import torch.backends.cudnn as cudnn
```

In [4]:

```
Hallo
```

In [5]:
```
from dataset import dataset
from AlexNet import AlexNet
```

In [6]:
```
trainloader, testloader, outputs, inputs = dataset('mnist')
```

```
| Preparing MNIST dataset...
Downloading http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz (http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz)
Downloading http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz (http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz)
Downloading http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz (http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz)
Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz (http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz)
Processing...
Done!
Output classes: 10
Input channels: 1
```

In [7]:
```
net = AlexNet(num_classes = outputs,inputs=inputs)
```

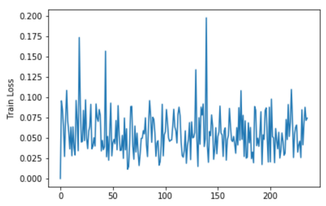
In [8]:
```
if use_cuda:
    net.cuda()
    net = torch.nn.DataParallel(net, device_ids=range(torch.cuda.device_count()))
```

In [9]:

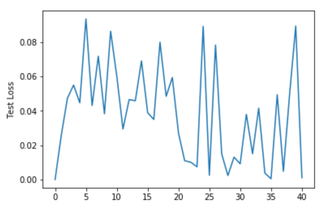
In [10]:

```
=> Training Epoch #1, LR=0.0010

D:\Dropbox\Dropbox\Master Theoretische Informatik\Very Deep Learning\WiSe 18_19\VDL_Python\Blatt 2\Very-Deep-Learning-CNN\train_test.py:31: UserWarning: invalid index of a 0-dim tensor. This will be an error in PyTorch 0.5. Use tensor.item() to convert a 0-dim tensor to a Python number
  train_loss += loss.data[0]
D:\Dropbox\Dropbox\Master Theoretische Informatik\Very Deep Learning\WiSe 18_19\VDL_Python\Blatt 2\Very-Deep-Learning-CNN\train_test.py:35: UserWarning: invalid index of a 0-dim tensor. This will be an error in PyTorch 0.5. Use tensor.item() to convert a 0-dim tensor to a Python number
  train_loss_stacked = np.append(train_loss_stacked, loss.data[0].cpu().numpy())
D:\Dropbox\Dropbox\Master Theoretische Informatik\Very Deep Learning\WiSe 18_19\VDL_Python\Blatt 2\Very-Deep-Learning-CNN\train_test.py:37: UserWarning: invalid index of a 0-dim tensor. This will be an error in PyTorch 0.5. Use tensor.item() to convert a 0-dim tensor to a Python number
  %(epoch, cf.num_epochs, loss.data[0], 100.*correct/total))

| Epoch [  1/ 10]               Loss: 0.1932 Acc@1: 68.000%

D:\Dropbox\Dropbox\Master Theoretische Informatik\Very Deep Learning\WiSe 18_19\VDL_Python\Blatt 2\Very-Deep-Learning-CNN\train_test.py:57: UserWarning: invalid index of a 0-dim tensor. This will be an error in PyTorch 0.5. Use tensor.item() to convert a 0-dim tensor to a Python number
  test_loss += loss.data[0]
D:\Dropbox\Dropbox\Master Theoretische Informatik\Very Deep Learning\WiSe 18_19\VDL_Python\Blatt 2\Very-Deep-Learning-CNN\train_test.py:61: UserWarning: invalid index of a 0-dim tensor. This will be an error in PyTorch 0.5. Use tensor.item() to convert a 0-dim tensor to a Python number
  test_loss_stacked = np.append(test_loss_stacked, loss.data[0].cpu().numpy())
D:\Dropbox\Dropbox\Master Theoretische Informatik\Very Deep Learning\WiSe 18_19\VDL_Python\Blatt 2\Very-Deep-Learning-CNN\train_test.py:66: UserWarning: invalid index of a 0-dim tensor. This will be an error in PyTorch 0.5. Use tensor.item() to convert a 0-dim tensor to a Python number
  print("\n| Validation Epoch #%d\t\t\tLoss: %.4f Acc@1: %.2f%%" % (epoch, loss.data[0], acc))

| Validation Epoch #1                   Loss: 0.0059 Acc@1: 93.00%
* Test results : Acc@1 = 93.00%
| Elapsed time : 0:02:44

=> Training Epoch #2, LR=0.0010
| Epoch [  2/ 10]               Loss: 0.1497 Acc@1: 94.000%

| Validation Epoch #2                   Loss: 0.0207 Acc@1: 96.00%
* Test results : Acc@1 = 96.00%
| Elapsed time : 0:05:27

=> Training Epoch #3, LR=0.0010
| Epoch [  3/ 10]               Loss: 0.1116 Acc@1: 96.000%

| Validation Epoch #3                   Loss: 0.0120 Acc@1: 96.00%
* Test results : Acc@1 = 96.00%
| Elapsed time : 0:08:08

=> Training Epoch #4, LR=0.0010
| Epoch [  4/ 10]               Loss: 0.0343 Acc@1: 97.000%

| Validation Epoch #4                   Loss: 0.0075 Acc@1: 98.00%
* Test results : Acc@1 = 98.00%
| Elapsed time : 0:10:51

=> Training Epoch #5, LR=0.0010
| Epoch [  5/ 10]               Loss: 0.0339 Acc@1: 97.000%

| Validation Epoch #5                   Loss: 0.0032 Acc@1: 97.00%
* Test results : Acc@1 = 98.00%
| Elapsed time : 0:13:34

=> Training Epoch #6, LR=0.0010
| Epoch [  6/ 10]               Loss: 0.0919 Acc@1: 97.000%

| Validation Epoch #6                   Loss: 0.0010 Acc@1: 97.00%
* Test results : Acc@1 = 98.00%
| Elapsed time : 0:16:19

=> Training Epoch #7, LR=0.0010
| Epoch [  7/ 10]               Loss: 0.0106 Acc@1: 97.000%

| Validation Epoch #7                   Loss: 0.0019 Acc@1: 98.00%
* Test results : Acc@1 = 98.00%
| Elapsed time : 0:19:03

=> Training Epoch #8, LR=0.0010
| Epoch [  8/ 10]               Loss: 0.0541 Acc@1: 98.000%

| Validation Epoch #8                   Loss: 0.0013 Acc@1: 98.00%
* Test results : Acc@1 = 98.00%
| Elapsed time : 0:21:48

=> Training Epoch #9, LR=0.0010
| Epoch [  9/ 10]               Loss: 0.0465 Acc@1: 98.000%

| Validation Epoch #9                   Loss: 0.0010 Acc@1: 98.00%
* Test results : Acc@1 = 98.00%
| Elapsed time : 0:24:32

=> Training Epoch #10, LR=0.0010
| Epoch [ 10/ 10]               Loss: 0.0743 Acc@1: 98.000%

| Validation Epoch #10                  Loss: 0.0010 Acc@1: 98.00%
* Test results : Acc@1 = 98.00%
| Elapsed time : 0:27:16
```

In [11]:
```
plt.plot(train_loss)
plt.ylabel('Train Loss')
```



In [12]:
```
plt.plot(test_loss)
plt.ylabel('Test Loss')
```



In [ ]:

In [ ]:

In [ ]: