show_results

November 3, 2016

1 Load model

1.1 Model

conv1_1 = new_conv_layer(image, [5, 5, 1 , 16], “conv1_1”) conv1_2 = new_conv_layer(conv1_1, [5, 5, 16, 16], “conv1_2”)
conv2_1 = new_conv_layer(conv1_2, [5, 5, 16, 16], “conv2_1”) conv2_2 = new_conv_layer(conv2_1, [5, 5, 16, 16], “conv2_2”)
gap = tf.reduce_mean(conv2_2, [1,2]) ccn = tf.reshape(gap,[-1,10,nb_CCN]) ccn = tf.reduce_mean(ccn, 2)

1.2 train params

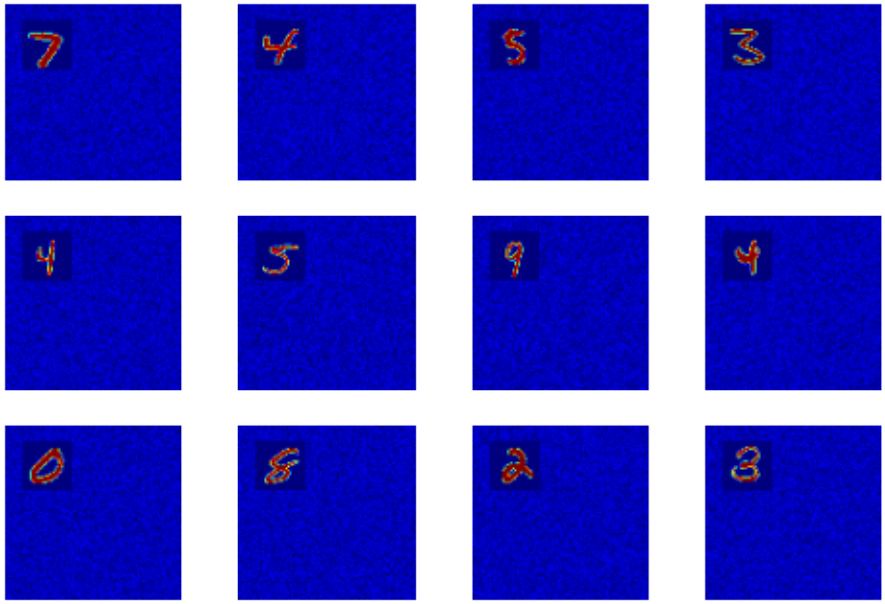
L2 on weights (5e-5) L1 on GAP

```
In [1]: import matplotlib.pyplot as plt
        import numpy as np
        import simple_model
        from simple_model import training_generator
        import utils


        simple_model=reload(simple_model)
        lr         = .005
        lr_decay   = .9
        back_size  = 100
        noise      = .1
        crop_pos   = (10,10)
        n_CCN      = 5
```

```
Exception AssertionError: AssertionError("Nesting violated for default stack of <ty
```

```
In [2]: """Plot training samples"""
        batch = utils.get_batch('train', im_size=back_size, noise=noise, crop_pos=c

        fig, axs = plt.subplots(3,4)
        for ax,img in zip([b for a in axs for b in a],batch[0]):
```

```
            ax.imshow(img.reshape(back_size,back_size), vmin=0, vmax=1)
            ax.set_axis_off()
        plt.show()
```



In [3]:
```
"""Perform training"""
gen = training_generator(lr=lr, lr_decay=lr_decay, back_size=back_size, noi
for _ in range(25):
    _,accs = gen.next()
    print "max acc so far : "+str(max(accs)*100)
```

```
***** EPOCH 0 ******
1-Loss on testset is 2.356416
1-Accuracy now is 11.49
2-Loss on testset is 2.301980
2-Accuracy now is 13.02
lr now is 0.00450
max acc so far : 11.49

***** EPOCH 1 ******
1-Loss on testset is 1.360155
1-Accuracy now is 75.79
2-Loss on testset is 1.361032
2-Accuracy now is 75.69
lr now is 0.00405
```

```
max acc so far : 75.79

***** EPOCH 2 ******
1-Loss on testset is 0.912301
1-Accuracy now is 88.79
2-Loss on testset is 0.916553
2-Accuracy now is 88.50
lr now is 0.00365
max acc so far : 88.79

***** EPOCH 3 ******
1-Loss on testset is 0.768343
1-Accuracy now is 90.11
2-Loss on testset is 0.766207
2-Accuracy now is 90.12
lr now is 0.00328
max acc so far : 90.11

***** EPOCH 4 ******
1-Loss on testset is 0.680467
1-Accuracy now is 90.77
2-Loss on testset is 0.673926
2-Accuracy now is 90.96
lr now is 0.00295
max acc so far : 90.77

***** EPOCH 5 ******
1-Loss on testset is 0.618553
1-Accuracy now is 91.27
2-Loss on testset is 0.616227
2-Accuracy now is 91.36
lr now is 0.00266
max acc so far : 91.27

***** EPOCH 6 ******
1-Loss on testset is 0.504258
1-Accuracy now is 95.03
2-Loss on testset is 0.500635
2-Accuracy now is 95.07
lr now is 0.00239
max acc so far : 95.03

***** EPOCH 7 ******
1-Loss on testset is 0.472787
1-Accuracy now is 95.69
2-Loss on testset is 0.469958
2-Accuracy now is 95.63
lr now is 0.00215
```

```
max acc so far : 95.69

***** EPOCH 8 ******
1-Loss on testset is 0.457466
1-Accuracy now is 95.85
2-Loss on testset is 0.448033
2-Accuracy now is 96.15
lr now is 0.00194
max acc so far : 95.85

***** EPOCH 9 ******
1-Loss on testset is 0.437950
1-Accuracy now is 95.90
2-Loss on testset is 0.442658
2-Accuracy now is 95.66
lr now is 0.00174
max acc so far : 95.9

***** EPOCH 10 ******
1-Loss on testset is 0.409138
1-Accuracy now is 96.80
2-Loss on testset is 0.402230
2-Accuracy now is 96.91
lr now is 0.00157
max acc so far : 96.8

***** EPOCH 11 ******
1-Loss on testset is 0.402402
1-Accuracy now is 96.80
2-Loss on testset is 0.399086
2-Accuracy now is 96.84
lr now is 0.00141
max acc so far : 96.8

***** EPOCH 12 ******
1-Loss on testset is 0.399370
1-Accuracy now is 97.21
2-Loss on testset is 0.396558
2-Accuracy now is 97.10
lr now is 0.00127
max acc so far : 97.21

***** EPOCH 13 ******
1-Loss on testset is 0.413590
1-Accuracy now is 96.19
2-Loss on testset is 0.400492
2-Accuracy now is 96.57
lr now is 0.00114
```

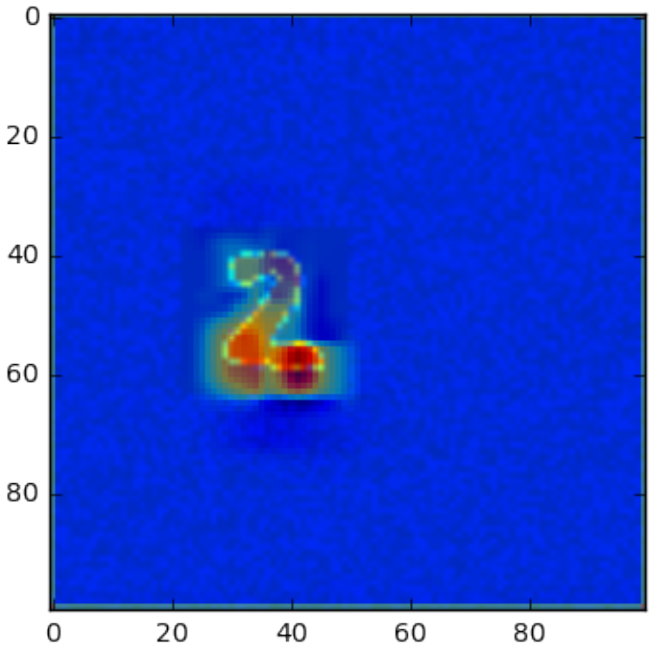
```
max acc so far : 97.21

***** EPOCH 14 ******
1-Loss on testset is 0.386078
1-Accuracy now is 96.76
2-Loss on testset is 0.387164
2-Accuracy now is 96.75
lr now is 0.00103
max acc so far : 97.21

***** EPOCH 15 ******
1-Loss on testset is 0.384180
1-Accuracy now is 97.20
2-Loss on testset is 0.380877
2-Accuracy now is 97.32
lr now is 0.00093
max acc so far : 97.21

***** EPOCH 16 ******
1-Loss on testset is 0.363440
1-Accuracy now is 97.24
2-Loss on testset is 0.355429
2-Accuracy now is 97.40
lr now is 0.00083
max acc so far : 97.24

***** EPOCH 17 ******
1-Loss on testset is 0.373458
1-Accuracy now is 97.15
2-Loss on testset is 0.362167
2-Accuracy now is 97.27
lr now is 0.00075
max acc so far : 97.24

***** EPOCH 18 ******
1-Loss on testset is 0.359458
1-Accuracy now is 97.58
2-Loss on testset is 0.352296
2-Accuracy now is 97.70
lr now is 0.00068
max acc so far : 97.58

***** EPOCH 19 ******
1-Loss on testset is 0.355177
1-Accuracy now is 97.52
2-Loss on testset is 0.348388
2-Accuracy now is 97.49
lr now is 0.00061
```

```
max acc so far : 97.58

***** EPOCH 20 ******
1-Loss on testset is 0.347565
1-Accuracy now is 97.45
2-Loss on testset is 0.339475
2-Accuracy now is 97.62
lr now is 0.00055
max acc so far : 97.58

***** EPOCH 21 ******
1-Loss on testset is 0.351418
1-Accuracy now is 97.66
2-Loss on testset is 0.346345
2-Accuracy now is 97.66
lr now is 0.00049
max acc so far : 97.66

***** EPOCH 22 ******
1-Loss on testset is 0.349502
1-Accuracy now is 97.56
2-Loss on testset is 0.346330
2-Accuracy now is 97.52
lr now is 0.00044
max acc so far : 97.66

***** EPOCH 23 ******
1-Loss on testset is 0.350790
1-Accuracy now is 97.73
2-Loss on testset is 0.343002
2-Accuracy now is 97.77
lr now is 0.00040
max acc so far : 97.73

***** EPOCH 24 ******
1-Loss on testset is 0.345823
1-Accuracy now is 97.66
2-Loss on testset is 0.336625
2-Accuracy now is 97.68
lr now is 0.00036
max acc so far : 97.73
```

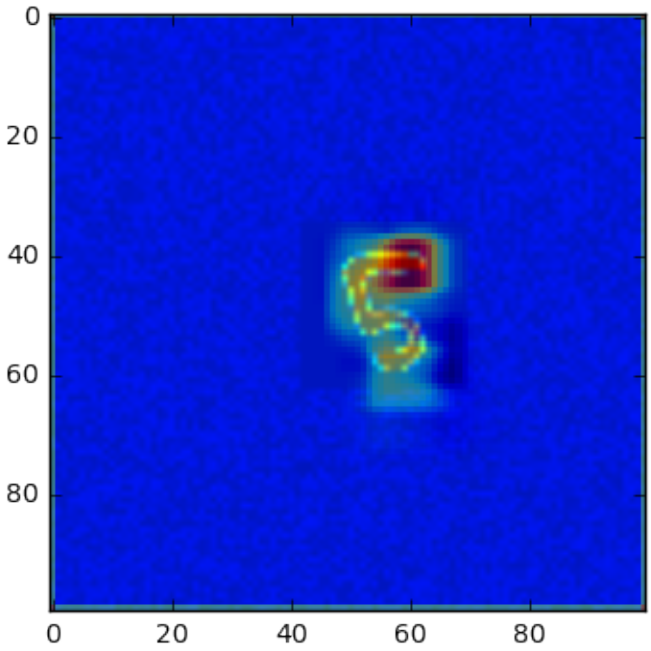
```
In [4]: back_size  = 100

        for _ in range(10):
            print '----------------'*5
            imgs, lbls = utils.get_batch('test', 1, back_size, .1).next()
```

```
simple_model.show_activation(imgs[0])
```

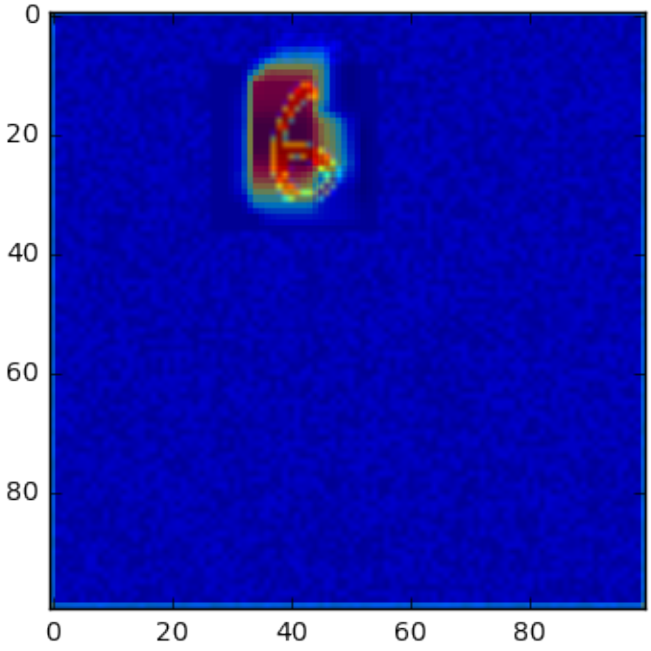
```
--------------------------------------------------------------------------------
prediction is : 2 with 7.963
```



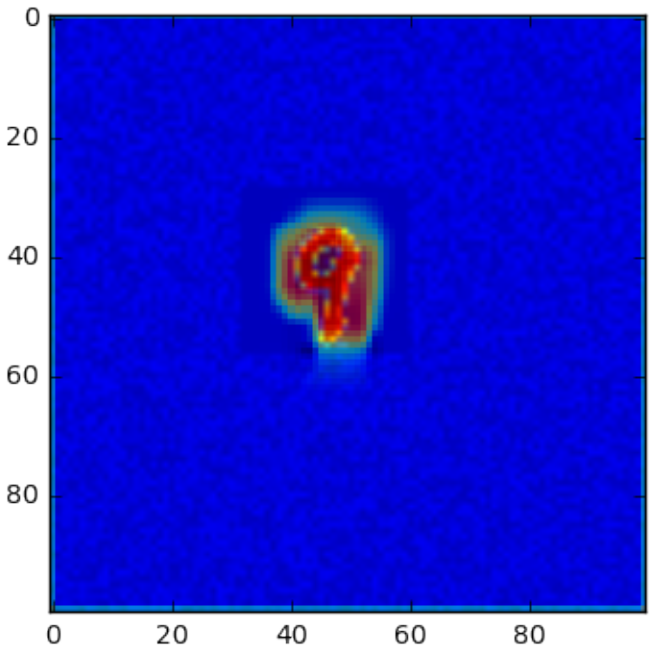

```
--------------------------------------------------------------------------------
prediction is : 5 with 12.150
```



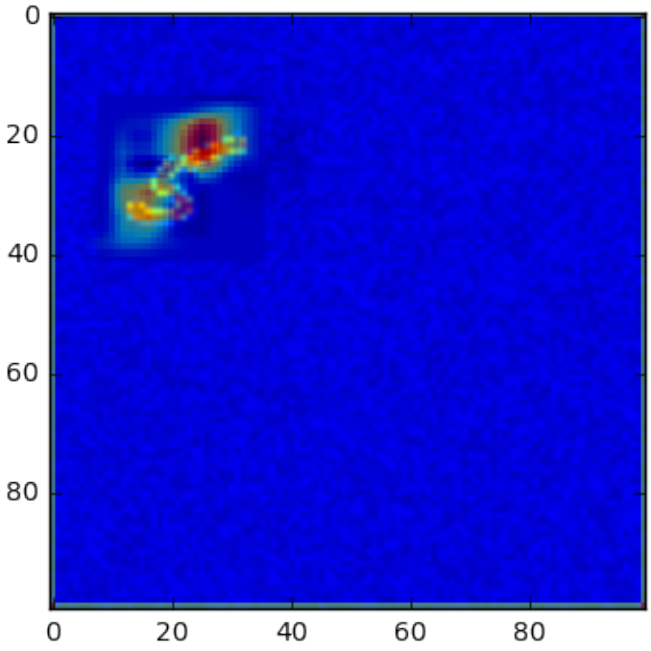
```
prediction is : 6 with 7.511
```



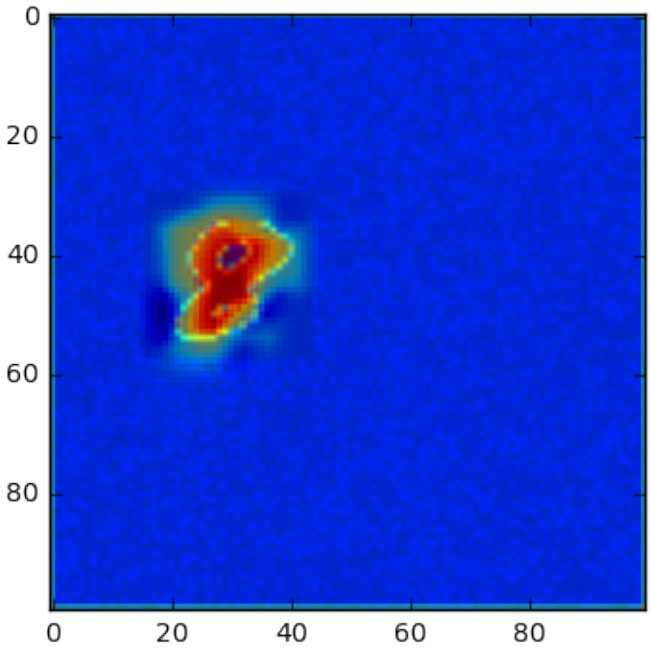
```
--------------------------------------------------------------------------------
prediction is : 9 with 8.456
```



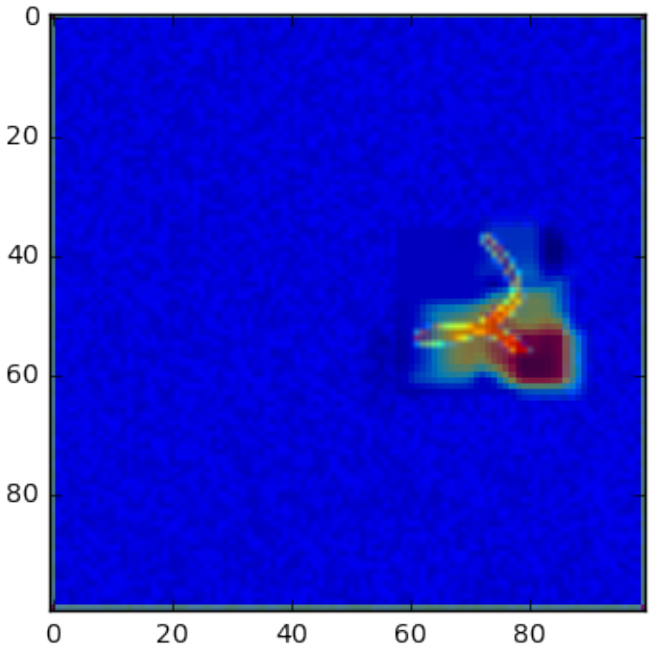

```
--------------------------------------------------------------------------------
prediction is : 5 with 9.257
```



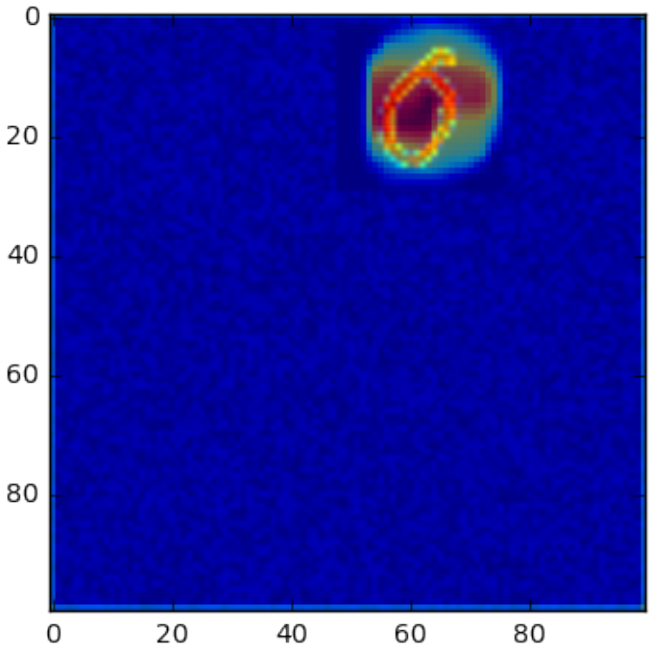
```
--------------------------------------------------------------------------------
prediction is : 8 with 10.372
```



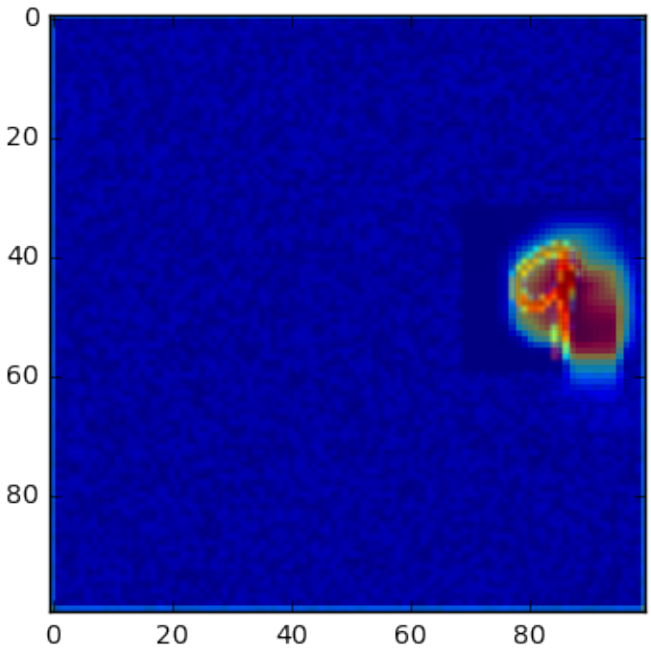

```
--------------------------------------------------------------------------------
prediction is : 2 with 6.469
```



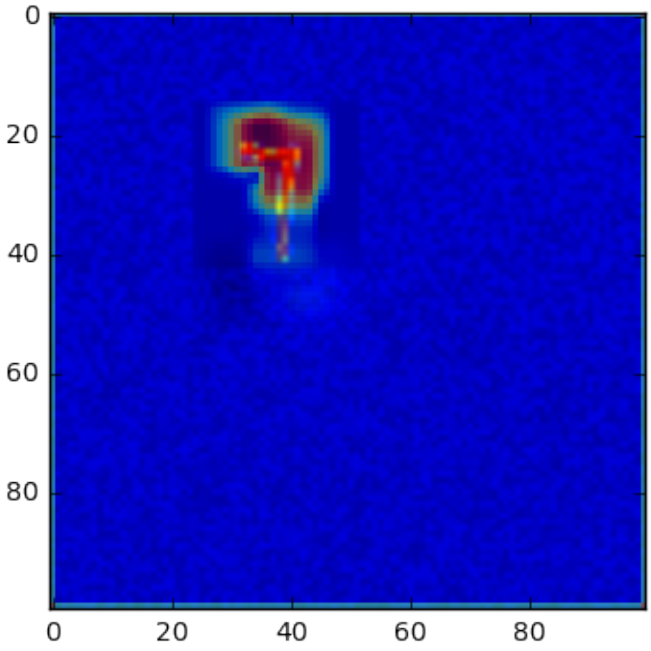
```
prediction is : 0 with 5.794
```



```
--------------------------------------------------------------------------------
prediction is : 9 with 8.243
```




```
--------------------------------------------------------------------------------
prediction is : 7 with 6.853
```



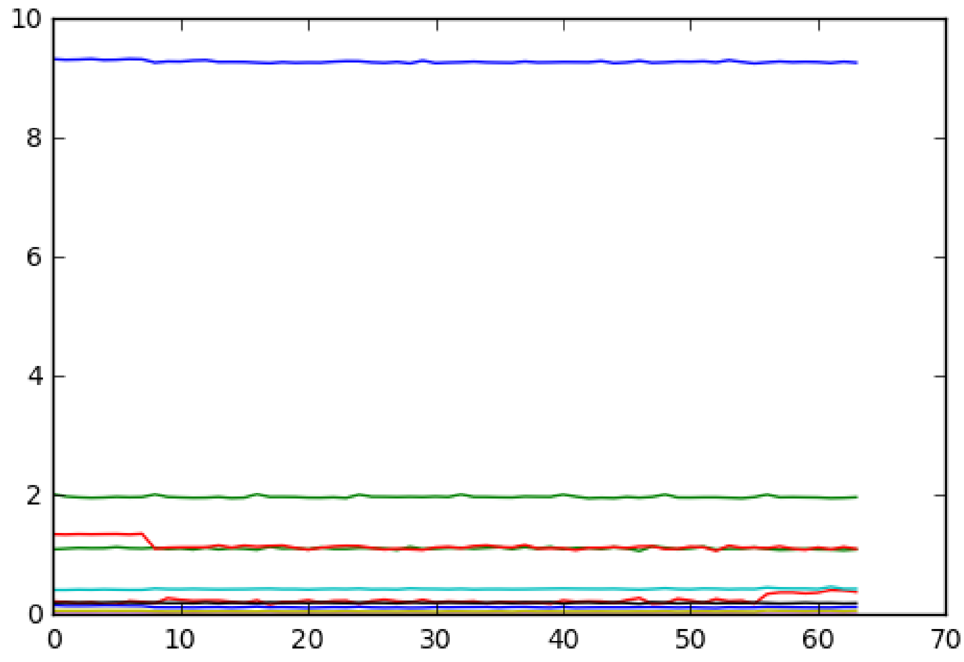
```
In [5]: accuracies = []
        for _ in range(100):
            digit,lbl  = utils.get_batch('test', 1, .1).next()
            digit      = digit[0].reshape(28,28)
            img        = np.random.random((100,100))*.1
            height     = img.shape[0]
            width      = img.shape[1]
            box_size   = 28
            step_size  = 10
            n_x_boxes  = (width -box_size)/step_size +1
            n_y_boxes  = (height-box_size)/step_size +1
            imgs       = np.tile(img, (n_x_boxes*n_y_boxes,1,1))

            for xx in range(0, n_x_boxes ):
                for yy in range(0, n_y_boxes ):
                    idx = xx*n_x_boxes+yy
                    x   = xx*step_size
                    y   = yy*step_size
                    imgs[idx, x:x+box_size, y:y+box_size ] = digit


            imgs  = imgs.reshape((-1,100,100,1))
            preds = simple_model.sess.run(simple_model.tf_out, feed_dict={simple_mo
            pred  = np.argmax(preds[-1])

            accuracy = sum(preds.argmax(axis=1) == lbl) / float(len(preds))
            accuracies.append(accuracy)

        print sum(accuracies)/len(accuracies)
```

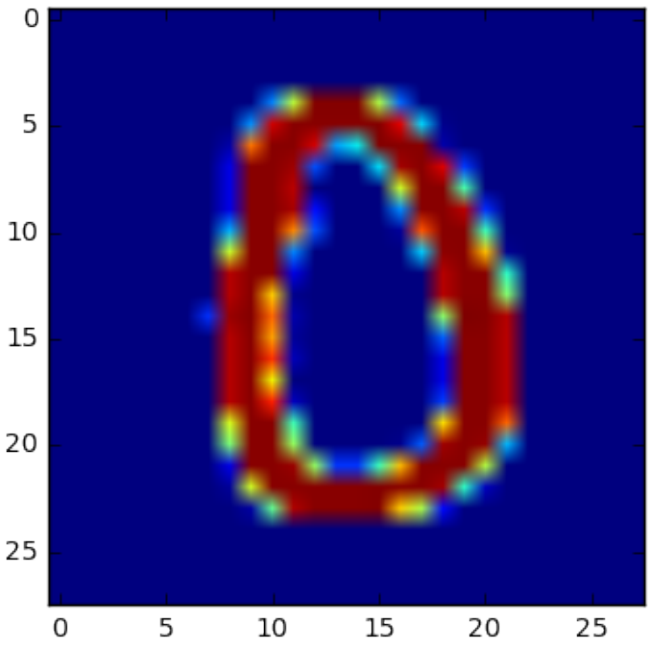
```
1.0
```

```
In [6]: digit,lbl  = utils.get_batch('test', 1, .1).next()
        digit      = digit[0].reshape(28,28)
        img        = np.random.random((100,100))*.1
        height     = img.shape[0]
        width      = img.shape[1]
        box_size   = 28
        step_size  = 10
        n_x_boxes  = (width -box_size)/step_size +1
        n_y_boxes  = (height-box_size)/step_size +1
        imgs       = np.tile(img, (n_x_boxes*n_y_boxes,1,1))

        for xx in range(0, n_x_boxes ):
```

```
            for yy in range(0, n_y_boxes ):
                idx = xx*n_x_boxes+yy
                x   = xx*step_size
                y   = yy*step_size
                imgs[idx, x:x+box_size, y:y+box_size ] = digit
        lr_decay

        imgs  = imgs.reshape((-1,100,100,1))
        preds = simple_model.sess.run(simple_model.tf_out, feed_dict={simple_model.
        pred  = np.argmax(preds[-1])

        accuracy = sum(preds.argmax(axis=1) == lbl) / float(len(preds))

        print preds.argmax(axis=1)
        print accuracy*100
        plt.plot(preds)
        plt.show()

        plt.imshow(digit)
        plt.show()
```

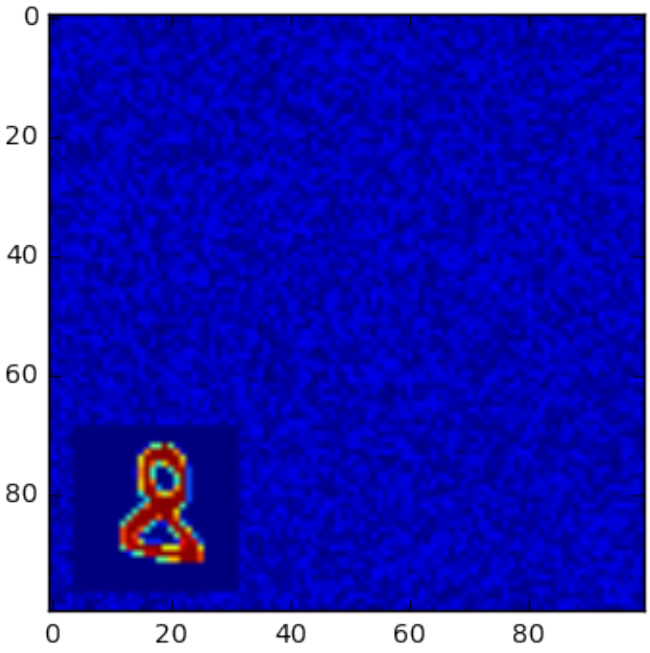
```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
100.0
```

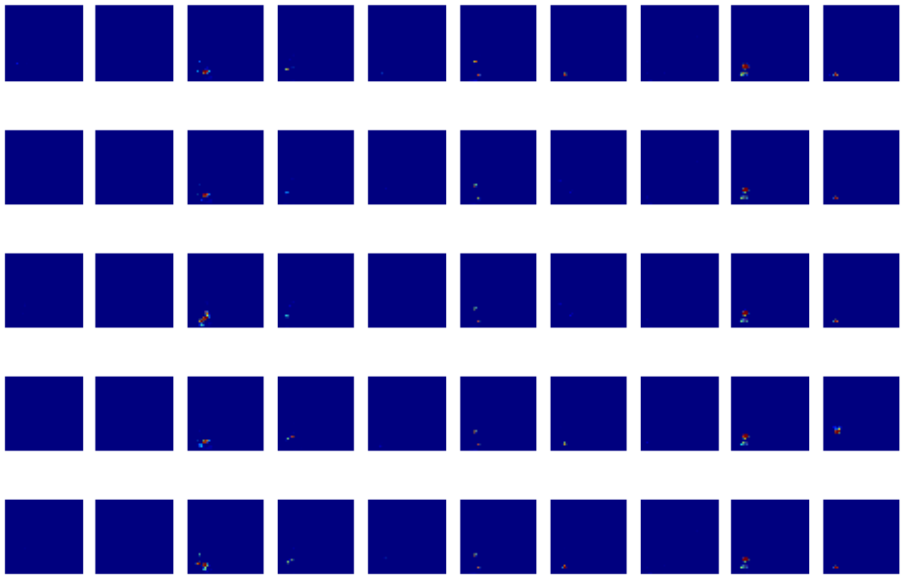




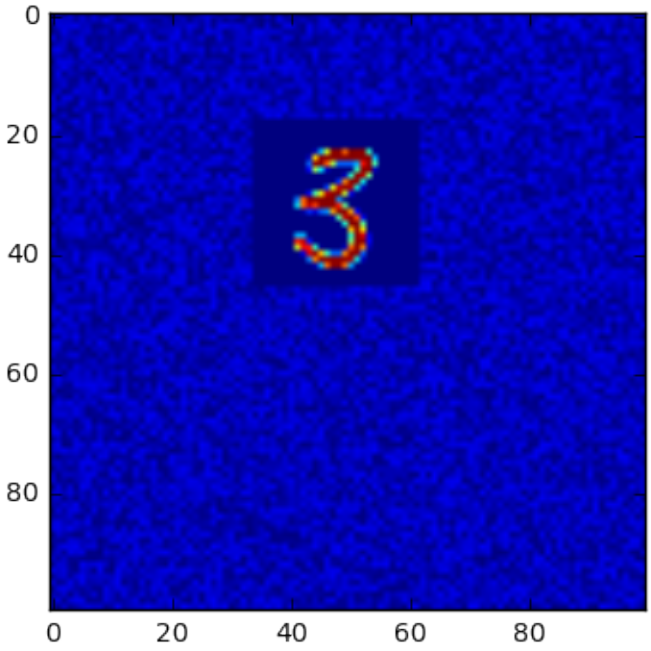

```
In [7]: for _ in range(10):
            simple_model.plot_classes_maps(100, n_CCN)
```

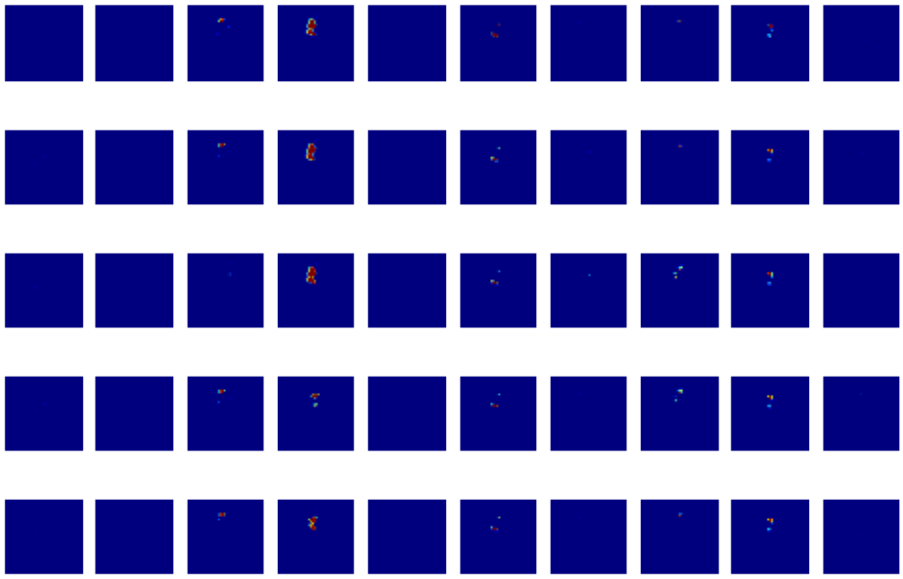
```
8
```



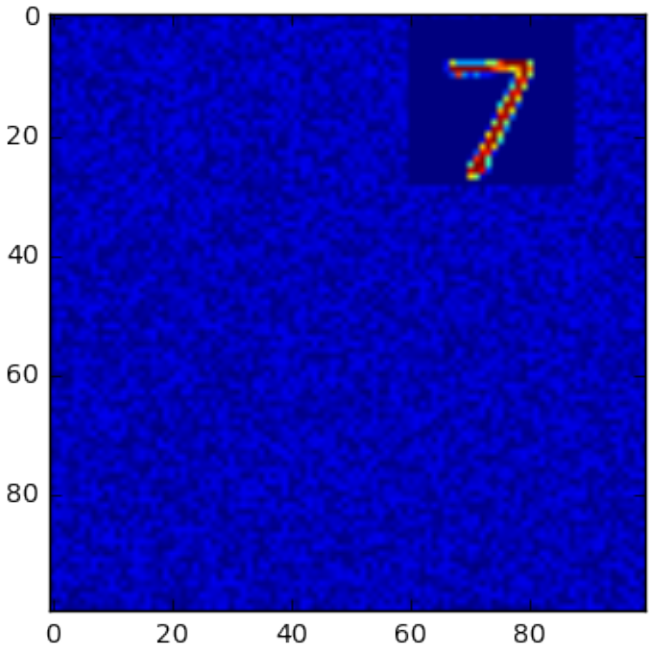


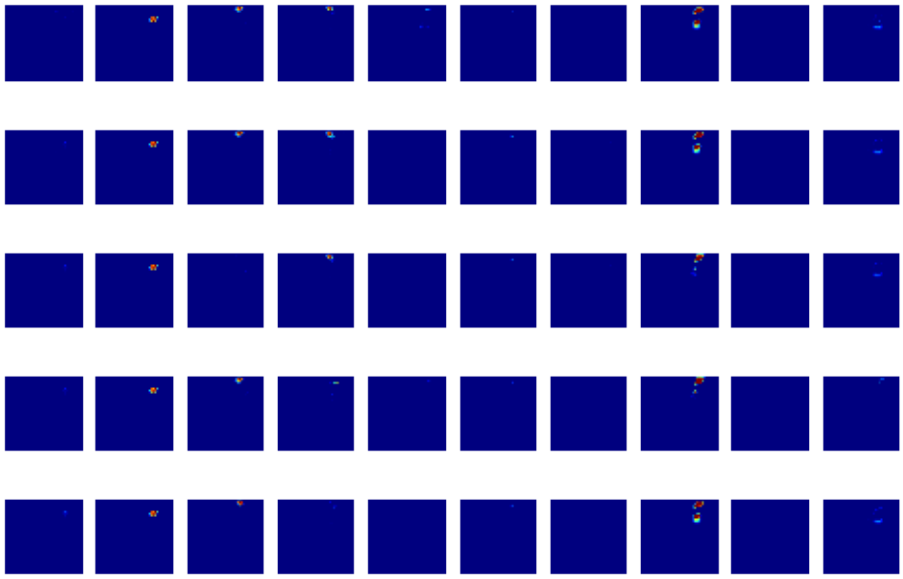
3



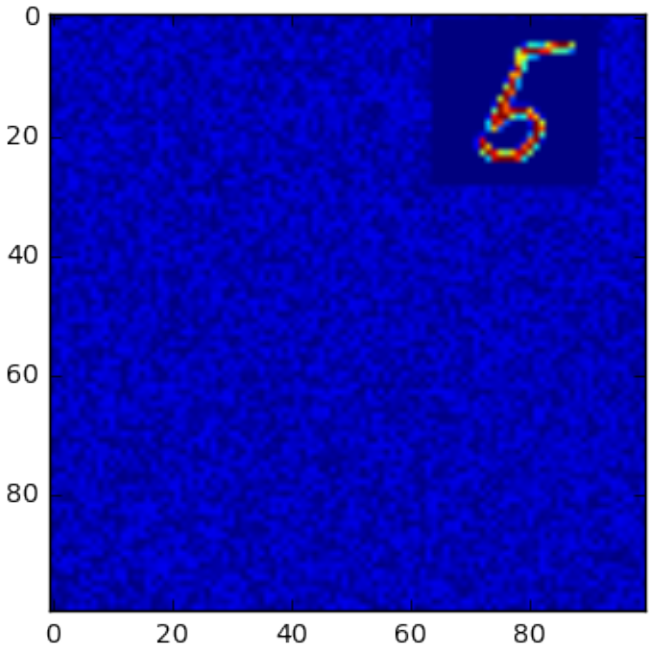


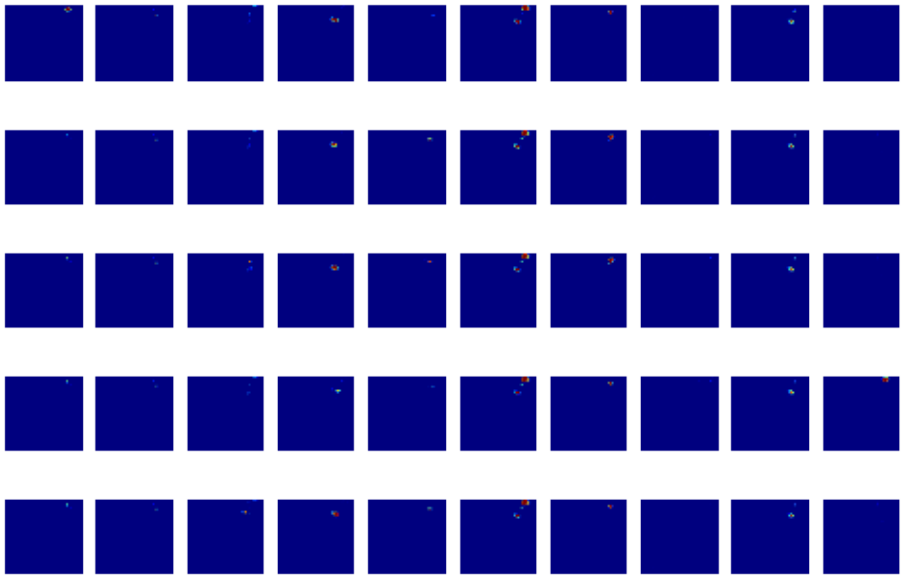
7



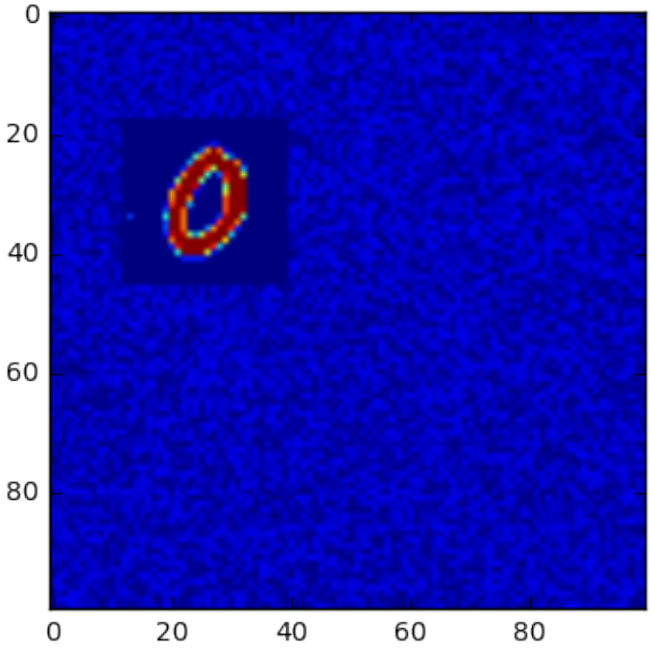


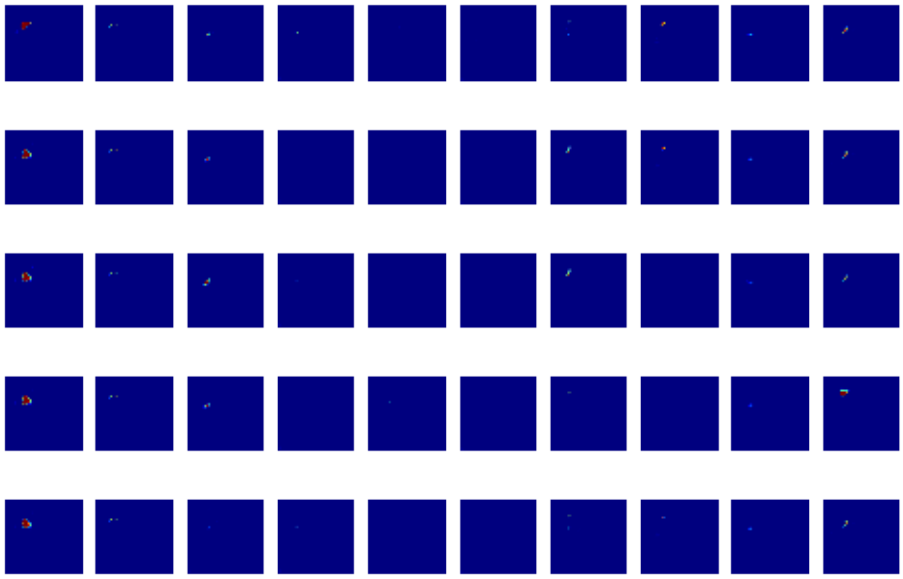
5



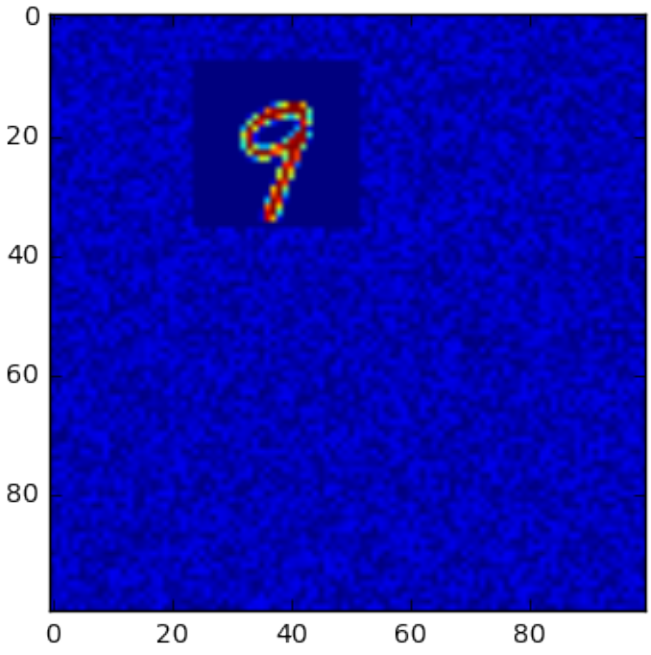


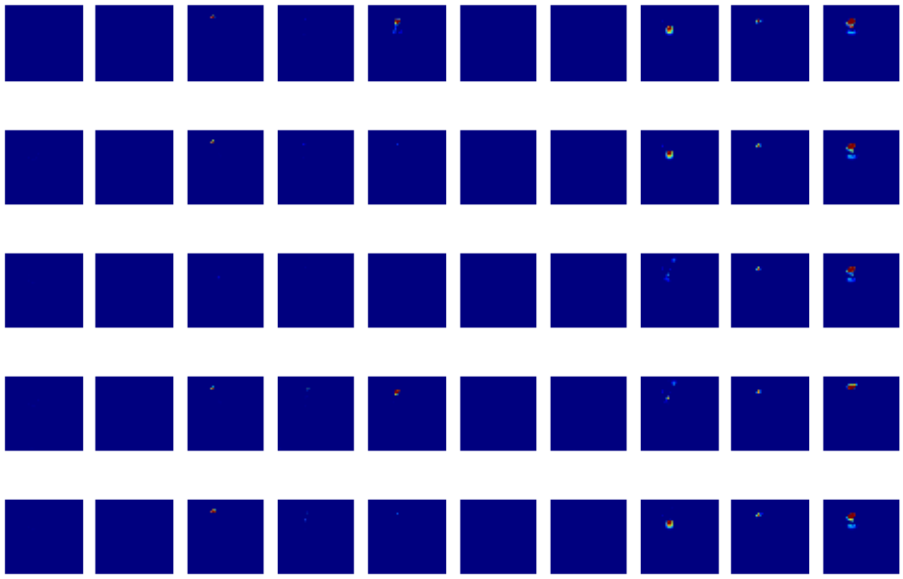
0



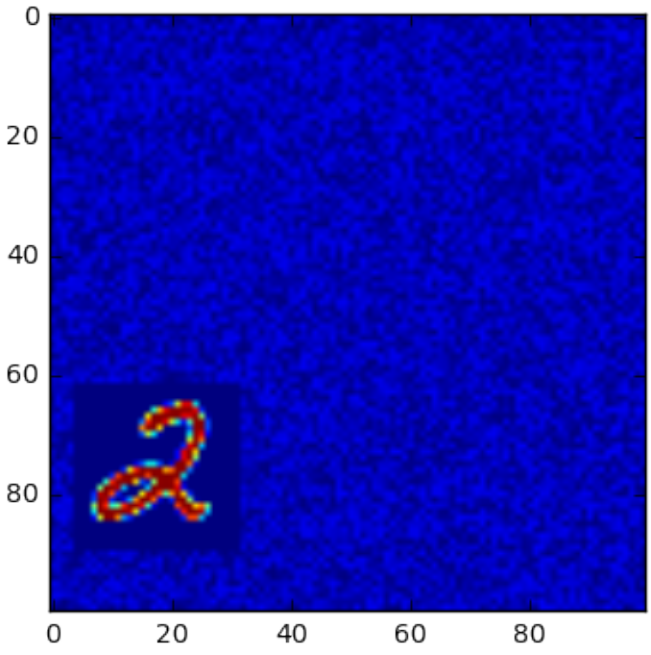


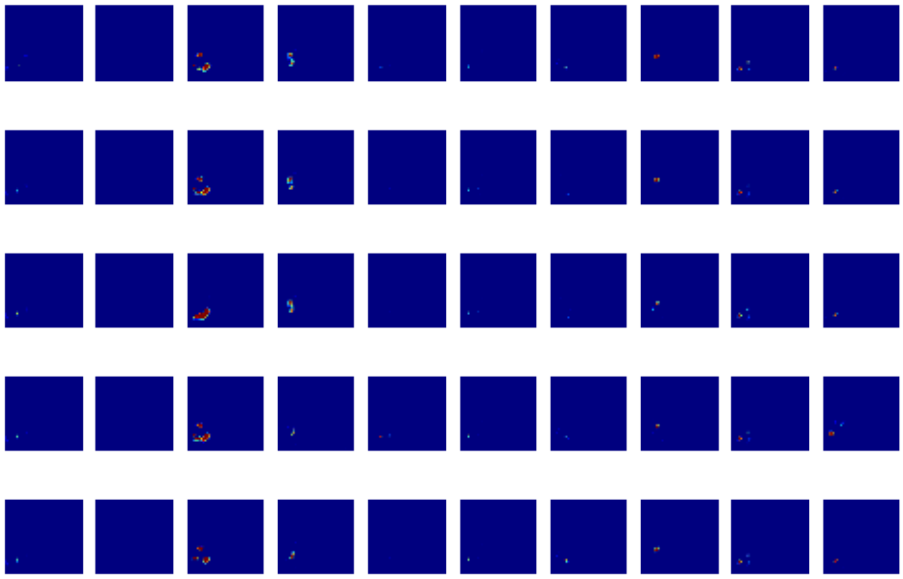
9



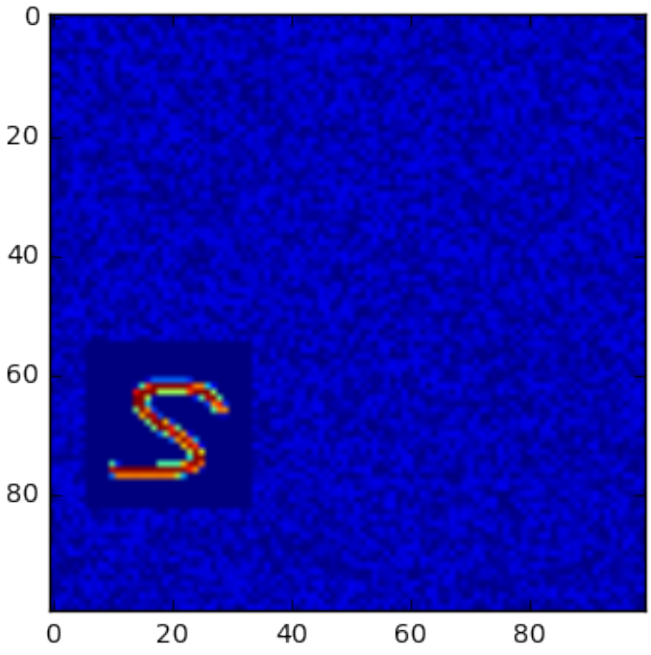


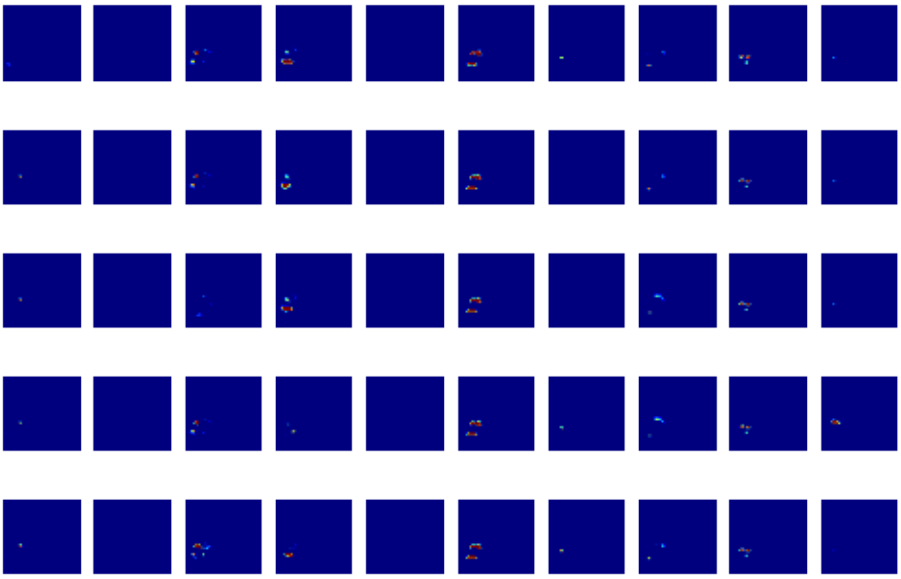
2



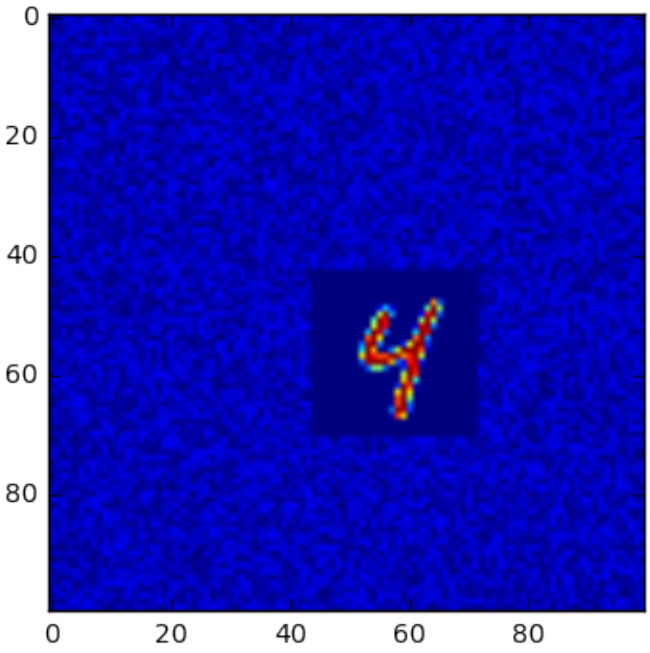


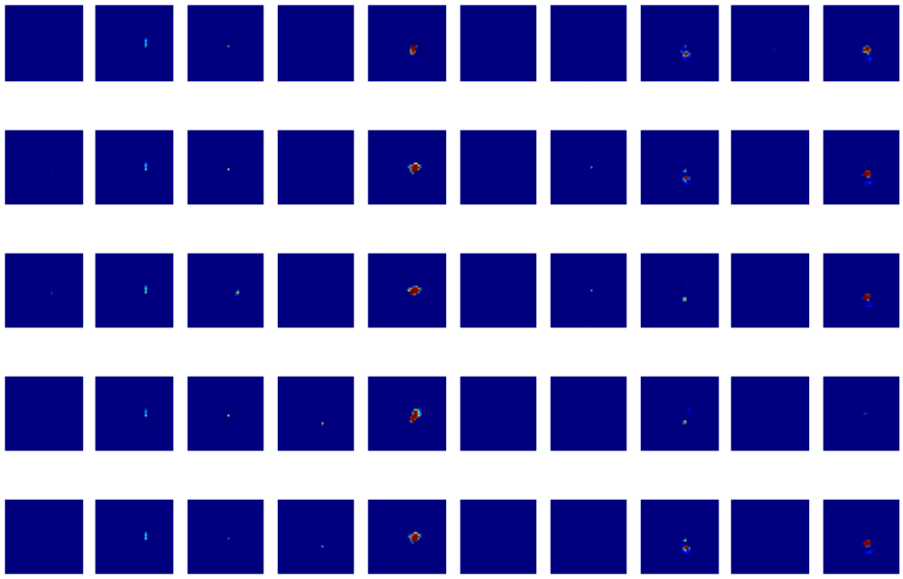
5



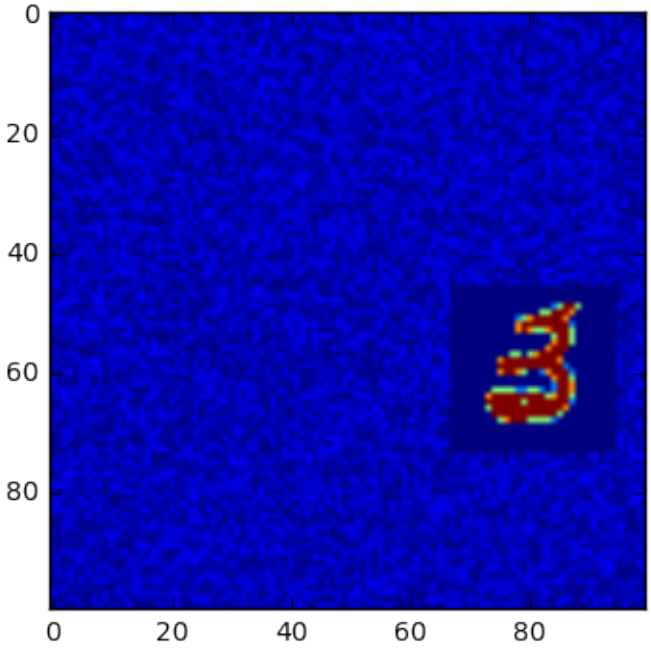


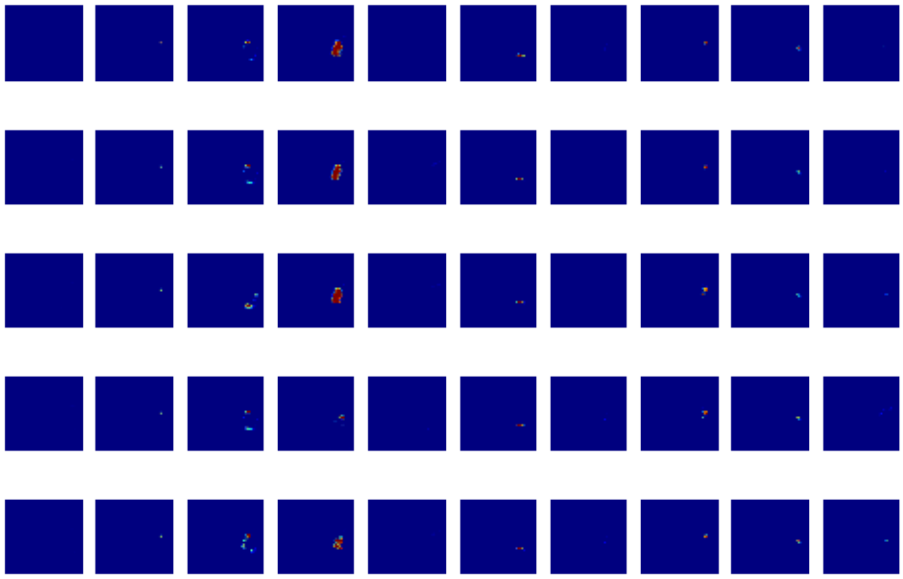
4





3





In [ ]: