show_results

November 3, 2016

1 Load model

1.1 Model

conv1_1 = new_conv_layer(image, [7, 7, 1 , 16], "conv1_1") conv1_2 = new_conv_layer(conv1_1, [7, 7, 16, 16], "conv1_2")

conv2_1 = new_conv_layer(conv1_2, [7, 7, 16, 16], "conv2_1") conv2_2 = new_conv_layer(conv2_1, [7, 7, 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

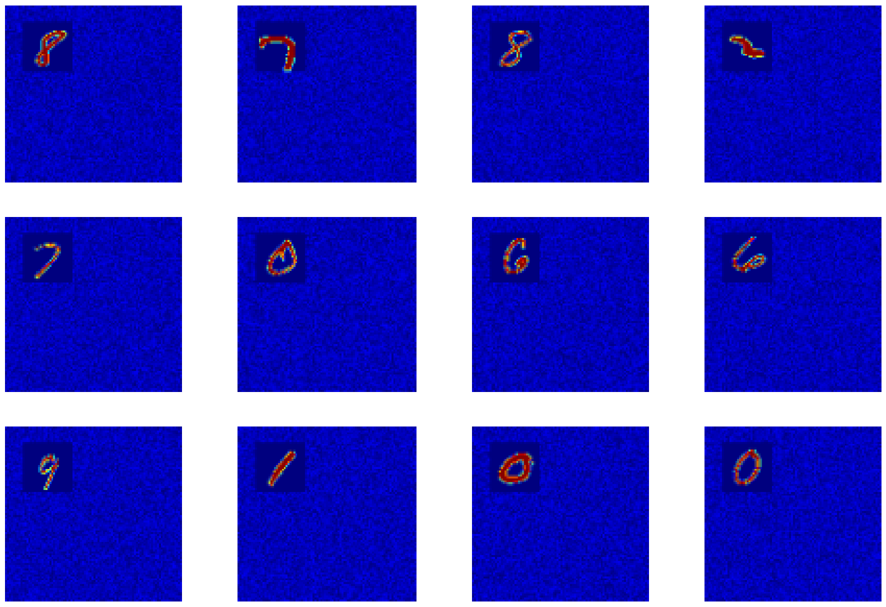
L2 on weights (5e-5) L1 on GAP

```
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 2.699204
1-Accuracy now is 25.60
2-Loss on testset is 2.363209
2-Accuracy now is 25.06
lr now is 0.00450
max acc so far : 25.6

***** EPOCH 1 ******
1-Loss on testset is 1.241657
1-Accuracy now is 76.80
2-Loss on testset is 1.231272
2-Accuracy now is 77.27
lr now is 0.00405
```

```
max acc so far : 76.8

***** EPOCH 2 ******
1-Loss on testset is 0.565248
1-Accuracy now is 92.73
2-Loss on testset is 0.538597
2-Accuracy now is 93.19
lr now is 0.00365
max acc so far : 92.73

***** EPOCH 3 ******
1-Loss on testset is 0.501866
1-Accuracy now is 93.03
2-Loss on testset is 0.495431
2-Accuracy now is 93.17
lr now is 0.00328
max acc so far : 93.03

***** EPOCH 4 ******
1-Loss on testset is 0.404001
1-Accuracy now is 95.81
2-Loss on testset is 0.393106
2-Accuracy now is 96.34
lr now is 0.00295
max acc so far : 95.81

***** EPOCH 5 ******
1-Loss on testset is 0.309083
1-Accuracy now is 98.09
2-Loss on testset is 0.308863
2-Accuracy now is 98.13
lr now is 0.00266
max acc so far : 98.09

***** EPOCH 6 ******
1-Loss on testset is 0.308618
1-Accuracy now is 97.63
2-Loss on testset is 0.298956
2-Accuracy now is 97.89
lr now is 0.00239
max acc so far : 98.09

***** EPOCH 7 ******
1-Loss on testset is 0.322913
1-Accuracy now is 97.38
2-Loss on testset is 0.321539
2-Accuracy now is 97.41
lr now is 0.00215
```

```
max acc so far : 98.09

***** EPOCH 8 ******
1-Loss on testset is 0.278519
1-Accuracy now is 98.16
2-Loss on testset is 0.275512
2-Accuracy now is 98.21
lr now is 0.00194
max acc so far : 98.16

***** EPOCH 9 ******
1-Loss on testset is 0.262065
1-Accuracy now is 98.40
2-Loss on testset is 0.262184
2-Accuracy now is 98.37
lr now is 0.00174
max acc so far : 98.4

***** EPOCH 10 ******
1-Loss on testset is 0.280382
1-Accuracy now is 97.59
2-Loss on testset is 0.282056
2-Accuracy now is 97.56
lr now is 0.00157
max acc so far : 98.4

***** EPOCH 11 ******
1-Loss on testset is 0.272341
1-Accuracy now is 97.78
2-Loss on testset is 0.274989
2-Accuracy now is 97.72
lr now is 0.00141
max acc so far : 98.4

***** EPOCH 12 ******
1-Loss on testset is 0.255198
1-Accuracy now is 98.31
2-Loss on testset is 0.255137
2-Accuracy now is 98.26
lr now is 0.00127
max acc so far : 98.4

***** EPOCH 13 ******
1-Loss on testset is 0.239455
1-Accuracy now is 98.58
2-Loss on testset is 0.238763
2-Accuracy now is 98.55
lr now is 0.00114
```

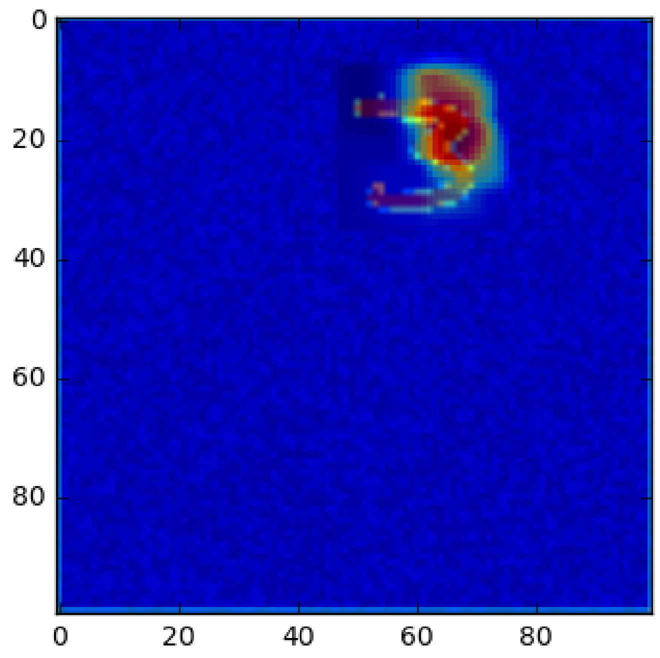
```
max acc so far : 98.58

***** EPOCH 14 ******
1-Loss on testset is 0.245529
1-Accuracy now is 98.61
2-Loss on testset is 0.241141
2-Accuracy now is 98.71
lr now is 0.00103
max acc so far : 98.61

***** EPOCH 15 ******
1-Loss on testset is 0.229240
1-Accuracy now is 98.78
2-Loss on testset is 0.226658
2-Accuracy now is 98.81
lr now is 0.00093
max acc so far : 98.78

***** EPOCH 16 ******
1-Loss on testset is 0.226836
1-Accuracy now is 98.58
2-Loss on testset is 0.224465
2-Accuracy now is 98.63
lr now is 0.00083
max acc so far : 98.78

***** EPOCH 17 ******
1-Loss on testset is 0.238768
1-Accuracy now is 98.46
2-Loss on testset is 0.233458
2-Accuracy now is 98.57
lr now is 0.00075
max acc so far : 98.78

***** EPOCH 18 ******
1-Loss on testset is 0.230852
1-Accuracy now is 98.54
2-Loss on testset is 0.229679
2-Accuracy now is 98.52
lr now is 0.00068
max acc so far : 98.78

***** EPOCH 19 ******
1-Loss on testset is 0.220806
1-Accuracy now is 98.89
2-Loss on testset is 0.219690
2-Accuracy now is 98.89
lr now is 0.00061
```

```
max acc so far : 98.89

***** EPOCH 20 ******
1-Loss on testset is 0.223807
1-Accuracy now is 98.86
2-Loss on testset is 0.220124
2-Accuracy now is 98.85
lr now is 0.00055
max acc so far : 98.89

***** EPOCH 21 ******
1-Loss on testset is 0.223892
1-Accuracy now is 98.97
2-Loss on testset is 0.220617
2-Accuracy now is 99.04
lr now is 0.00049
max acc so far : 98.97

***** EPOCH 22 ******
1-Loss on testset is 0.222955
1-Accuracy now is 98.75
2-Loss on testset is 0.217342
2-Accuracy now is 98.79
lr now is 0.00044
max acc so far : 98.97

***** EPOCH 23 ******
1-Loss on testset is 0.213430
1-Accuracy now is 98.91
2-Loss on testset is 0.211060
2-Accuracy now is 98.93
lr now is 0.00040
max acc so far : 98.97

***** EPOCH 24 ******
1-Loss on testset is 0.214400
1-Accuracy now is 98.83
2-Loss on testset is 0.211333
2-Accuracy now is 98.79
lr now is 0.00036
max acc so far : 98.97
```

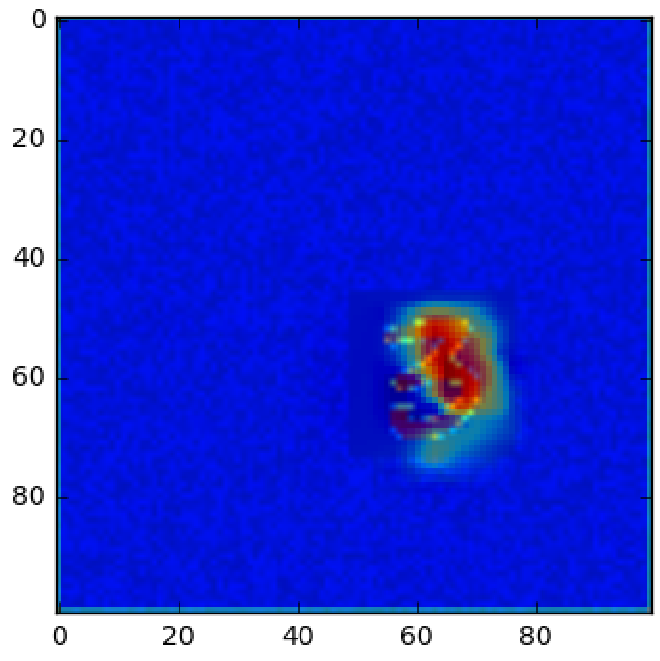
```
In [4]: back_size  = 100

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

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

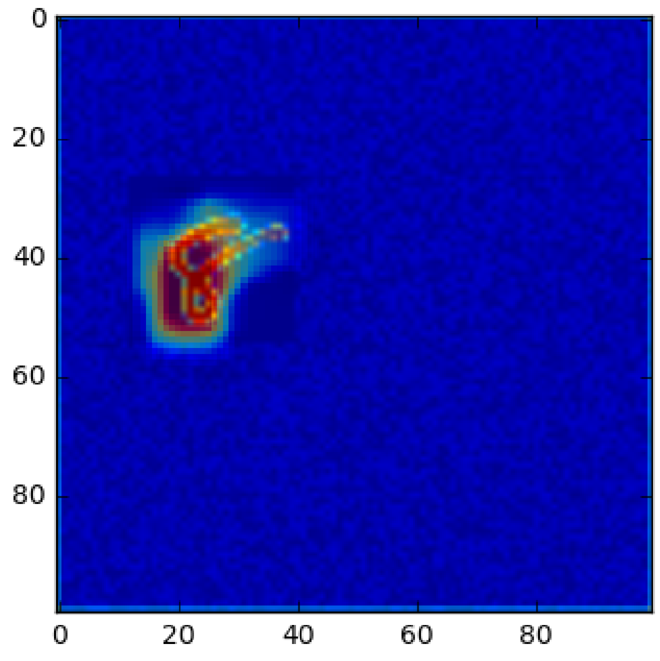
```
--------------------------------------------------------------------------------
prediction is : 3 with 13.017
```



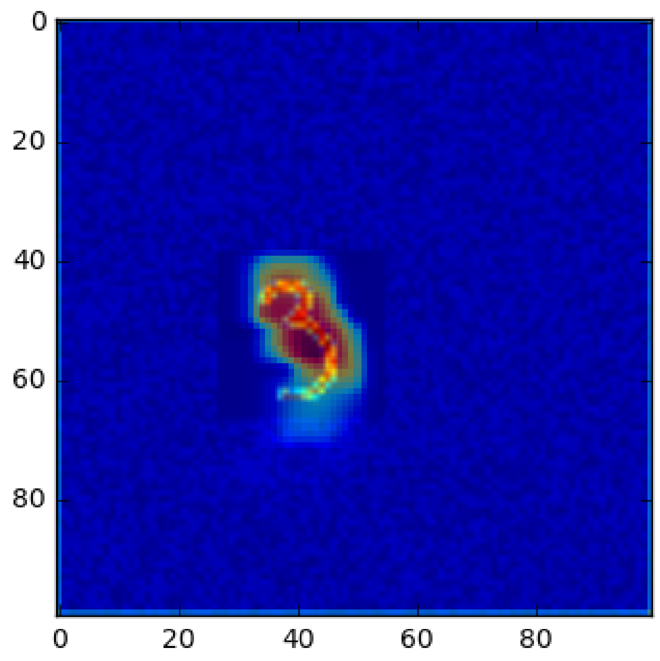
```
--------------------------------------------------------------------------------
prediction is : 3 with 10.138
```



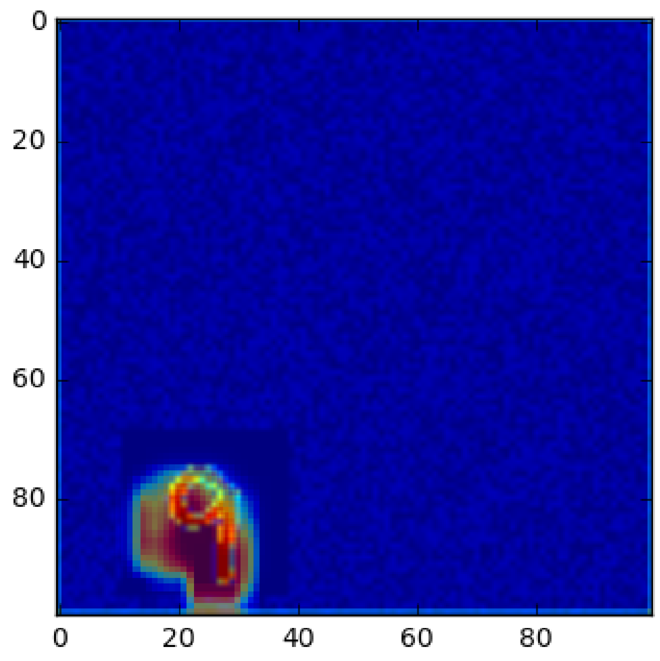
```
--------------------------------------------------------------------------------
prediction is : 8 with 12.652
```



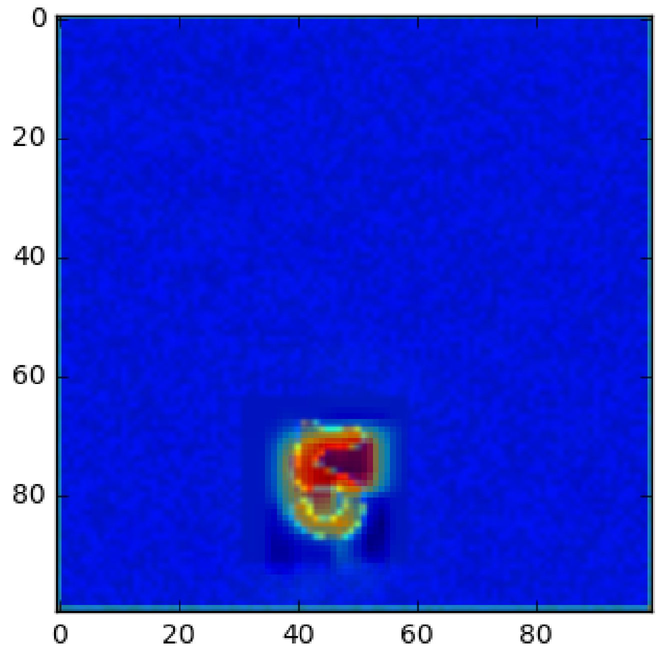
```
--------------------------------------------------------------------------------
prediction is : 3 with 8.147
```



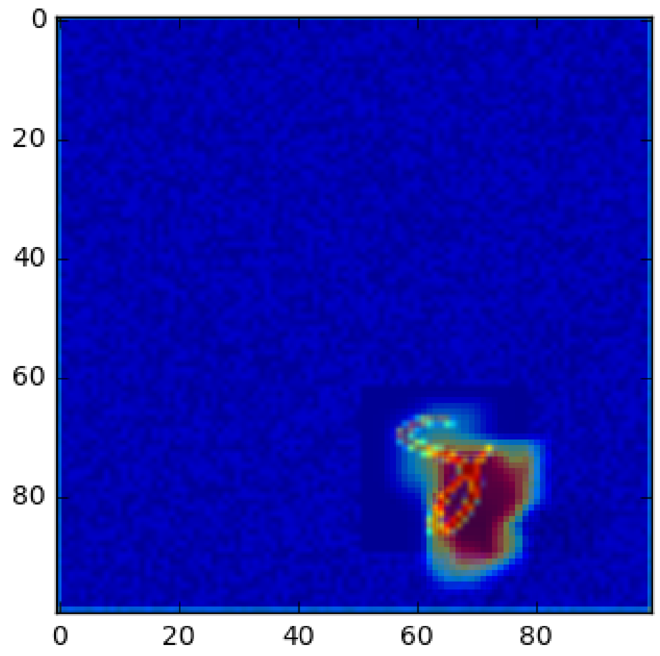
```
--------------------------------------------------------------------------------
prediction is : 9 with 11.427
```



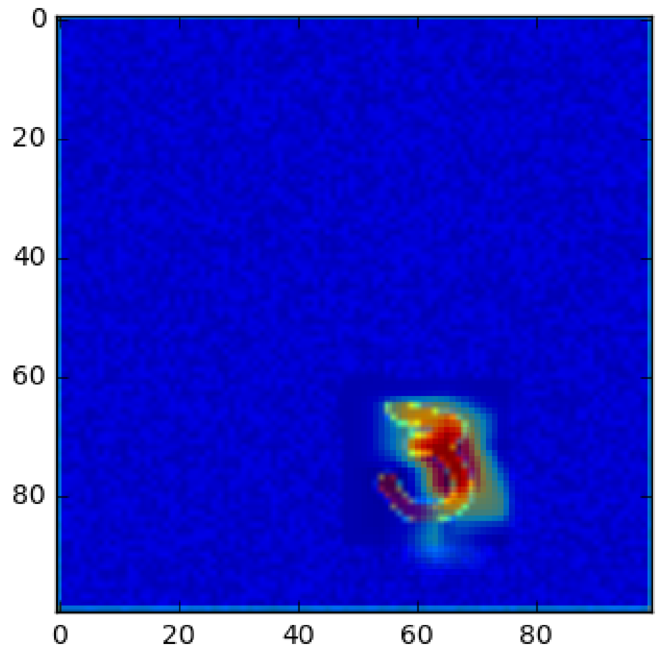
```
--------------------------------------------------------------------------------
prediction is : 5 with 8.572
```



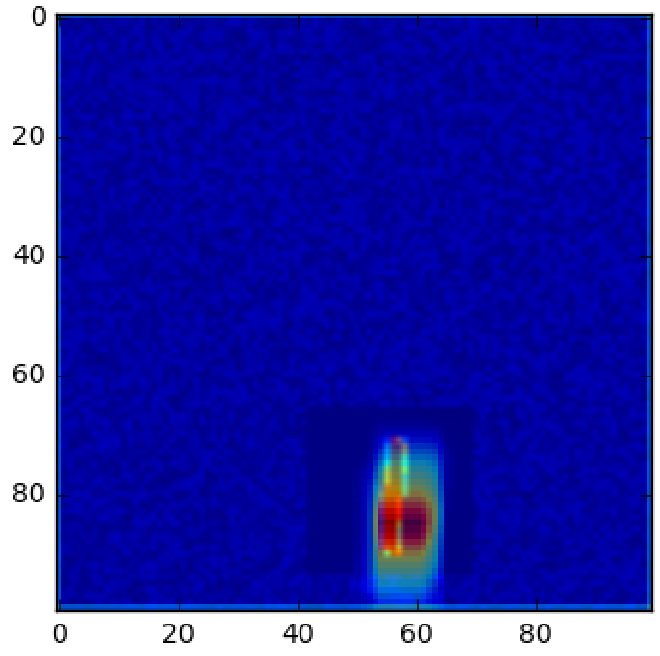
```
--------------------------------------------------------------------------------
prediction is : 8 with 5.059
```



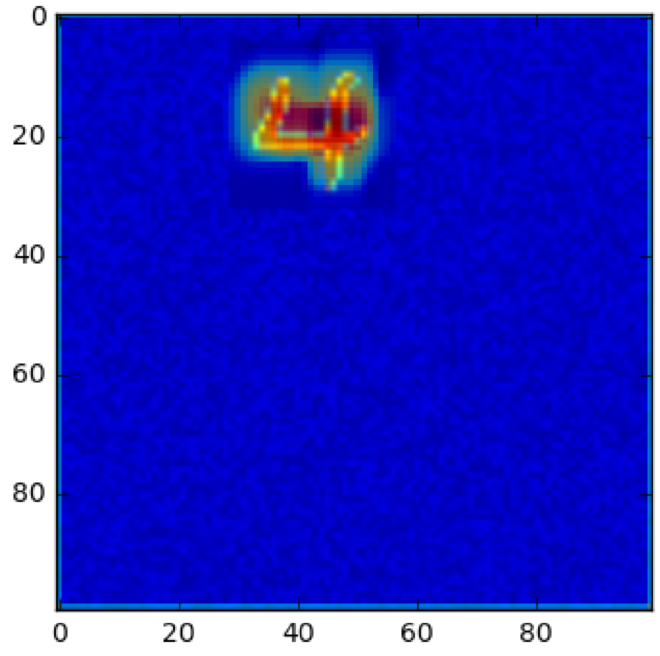
```
--------------------------------------------------------------------------------
prediction is : 3 with 10.695
```



```
--------------------------------------------------------------------------------
prediction is : 1 with 8.981
```



```
--------------------------------------------------------------------------------
prediction is : 4 with 13.470
```

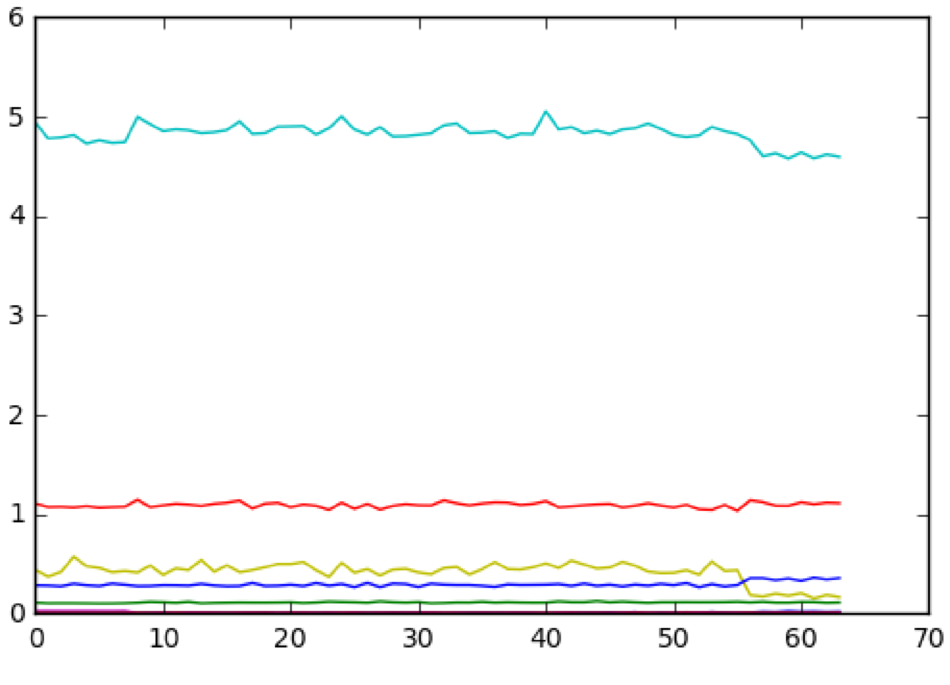


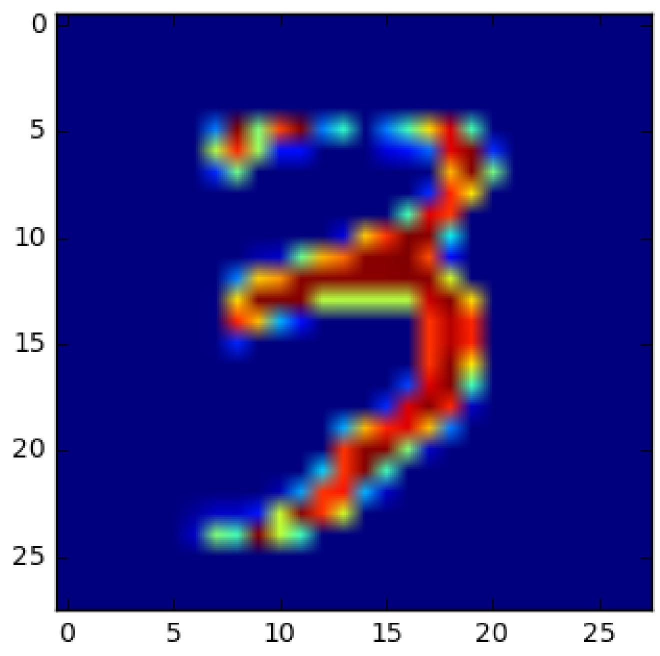
```
In [5]: accuracies = []
        for _ in range(100):
            digit,lbl   = utils.get_batch('test', 1, .1).next()
            digit       = digit[0].reshape(28,28)
            img         = np.random.random((100,100))*.1
            height      = img.shape[0]
            width       = img.shape[1]
            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()
```

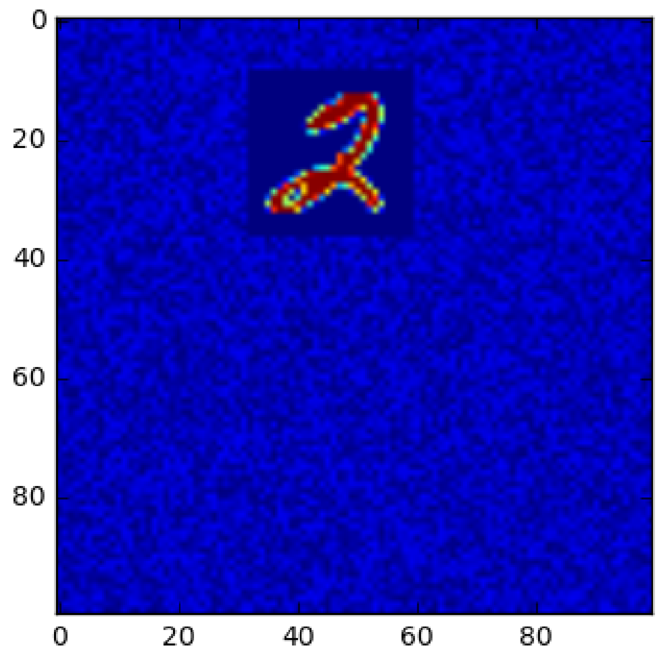
```
[3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3]
100.0
```

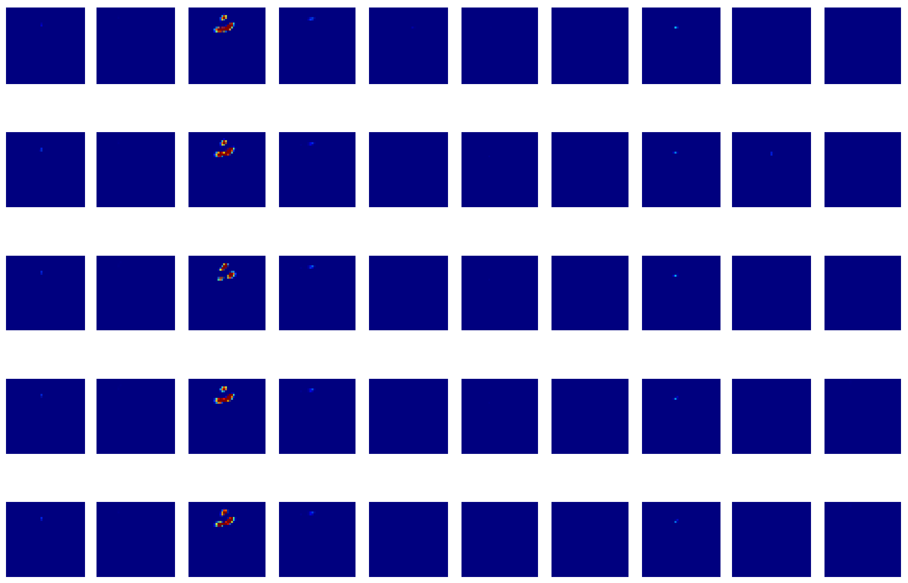




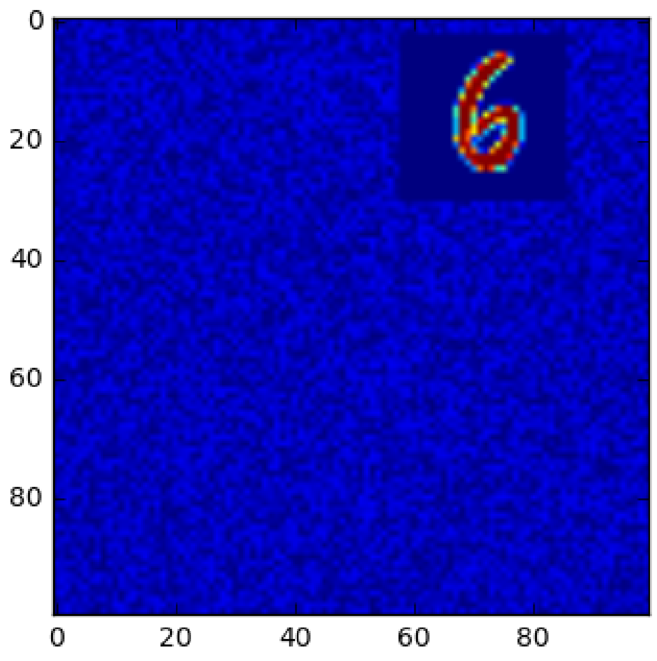
```
In [7]: for _ in range(10):
            simple_model.plot_classes_maps(100, n_CCN)
```

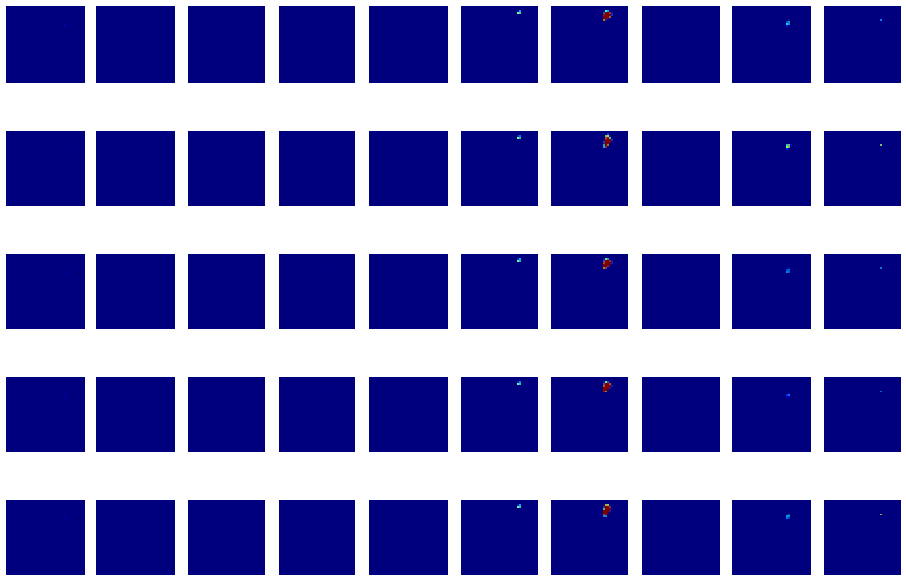
```
2
```



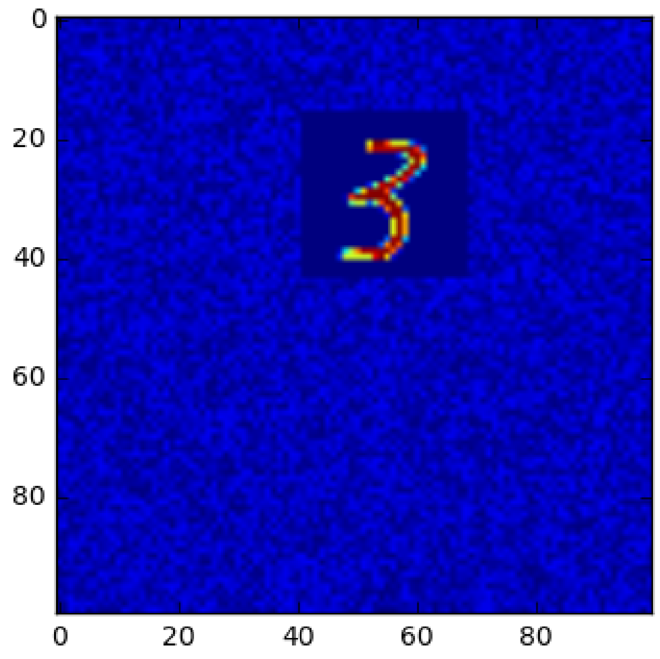


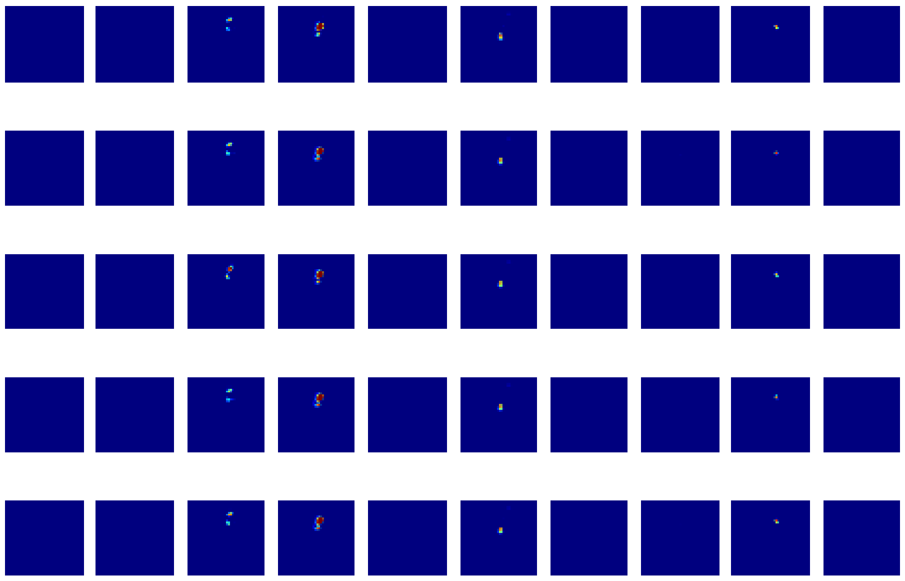
6



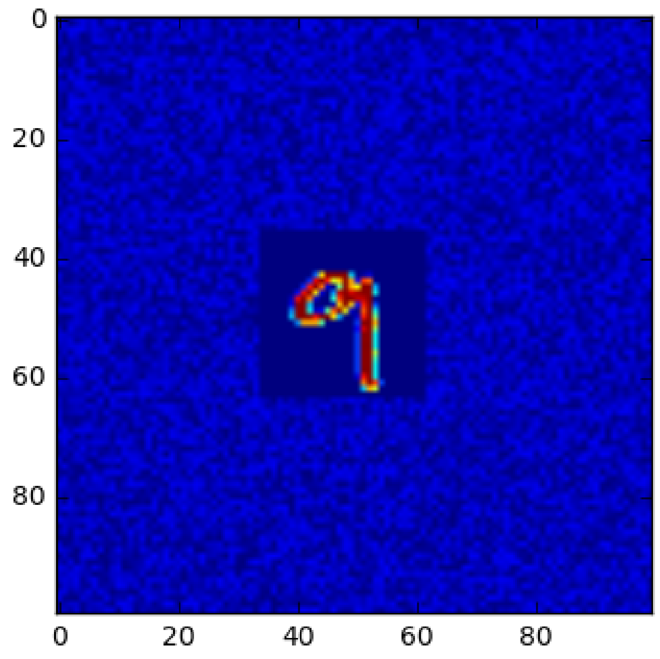


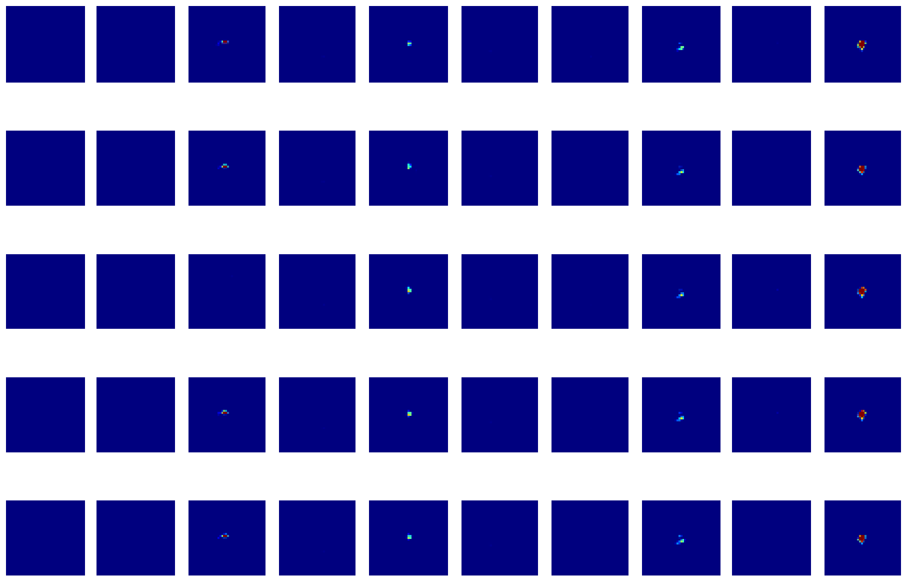
3



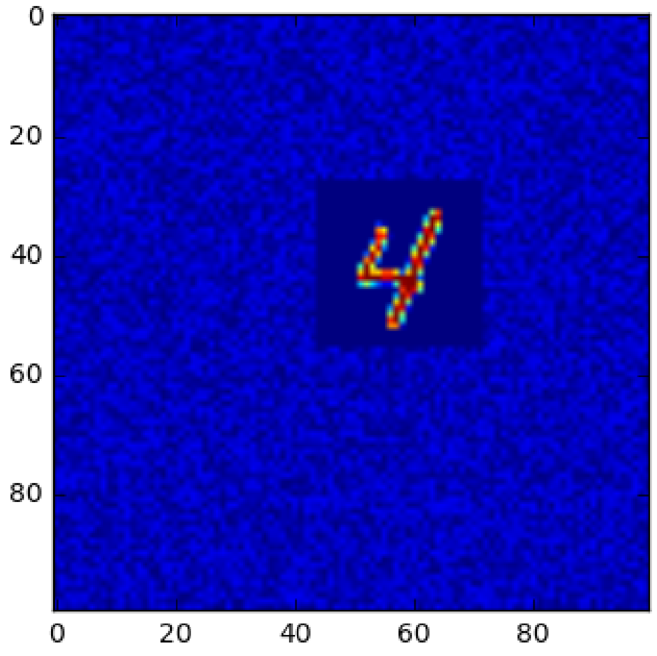


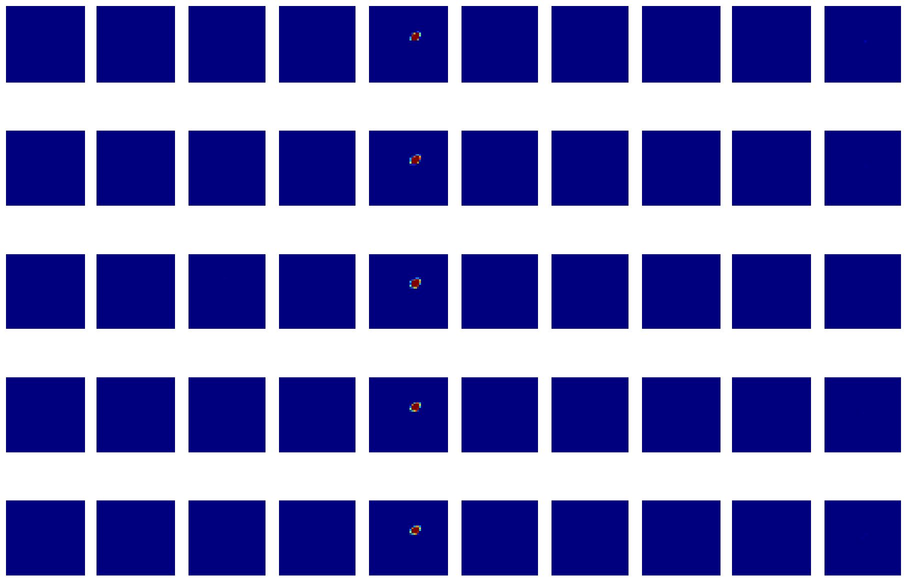
9



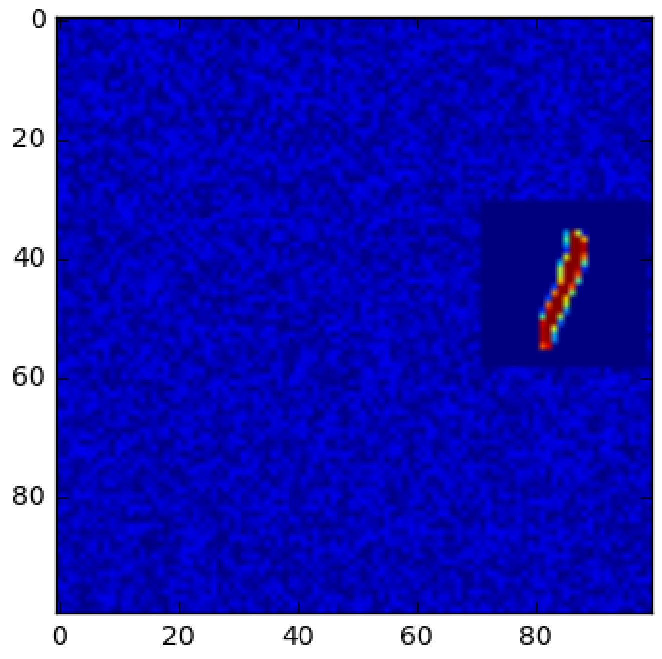


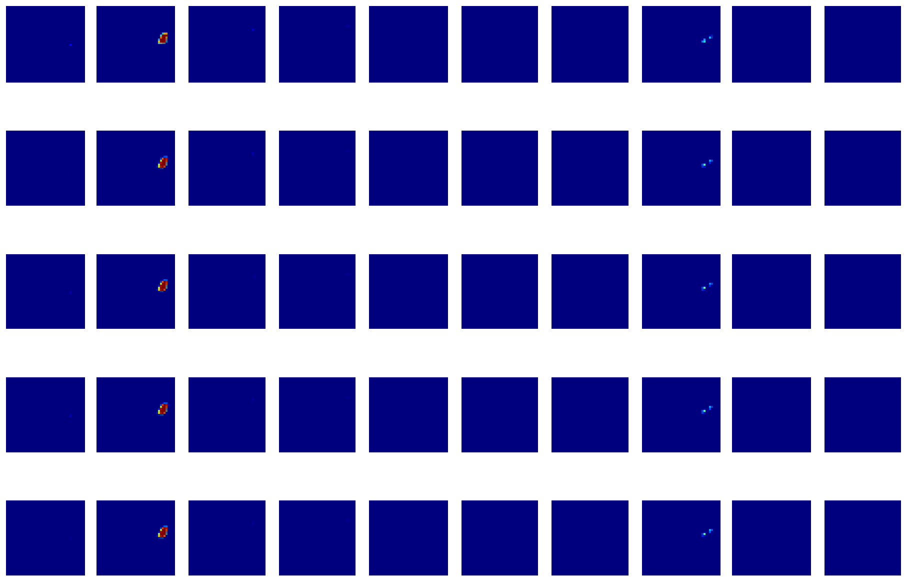
4



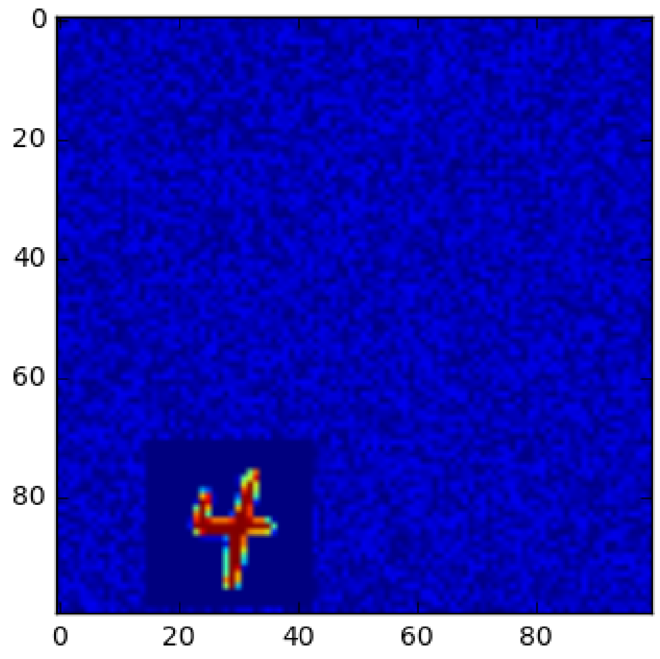


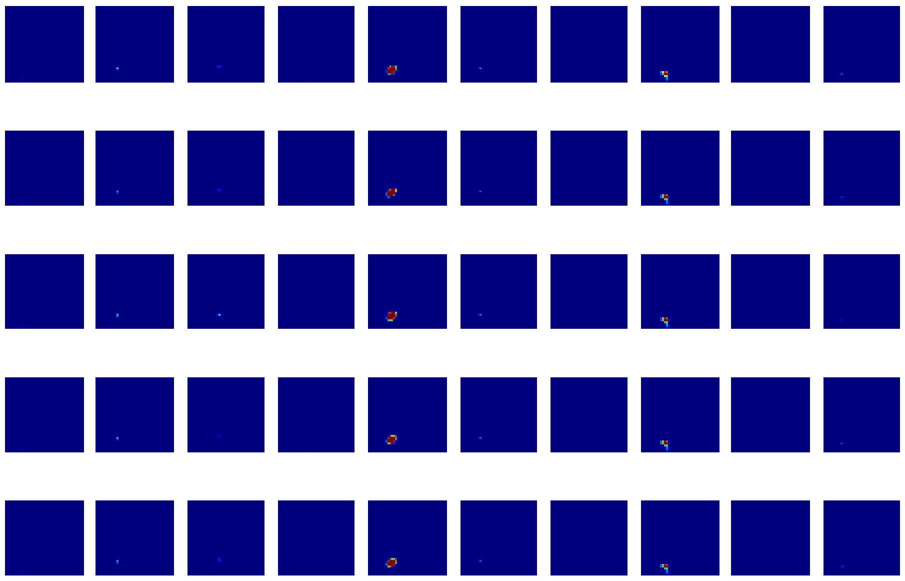
```
1
```



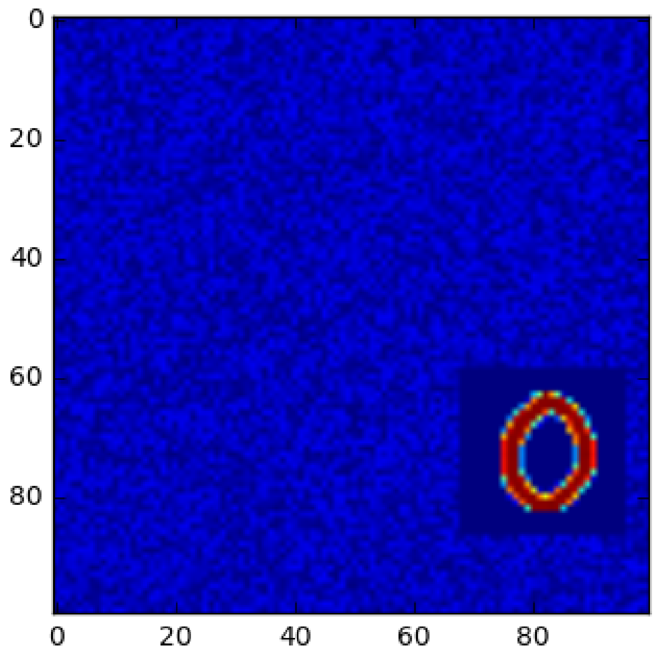


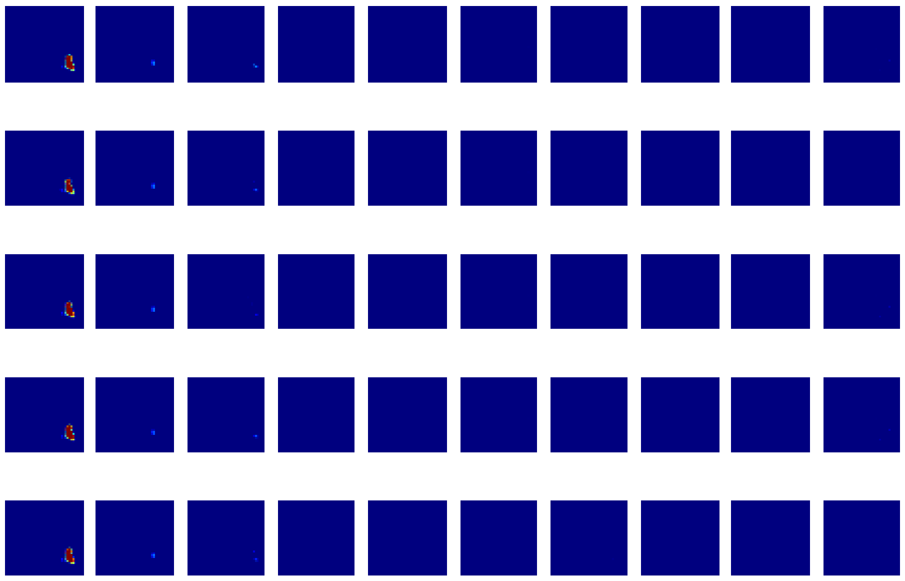
4



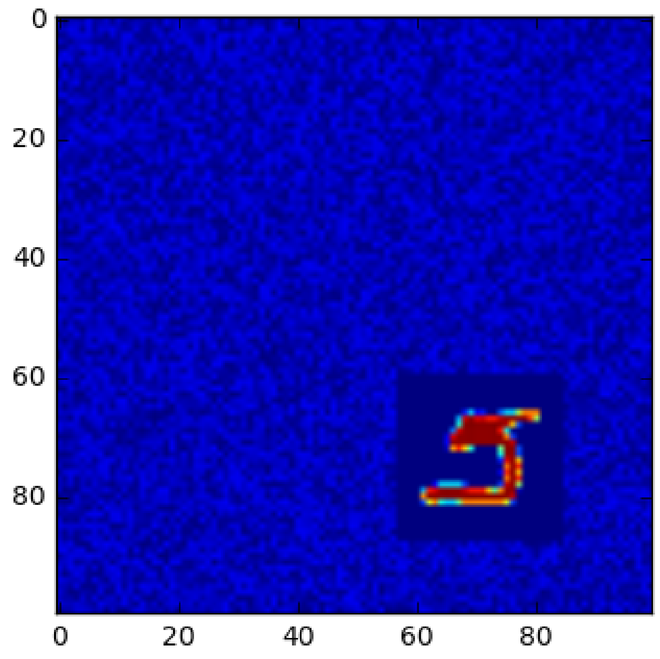


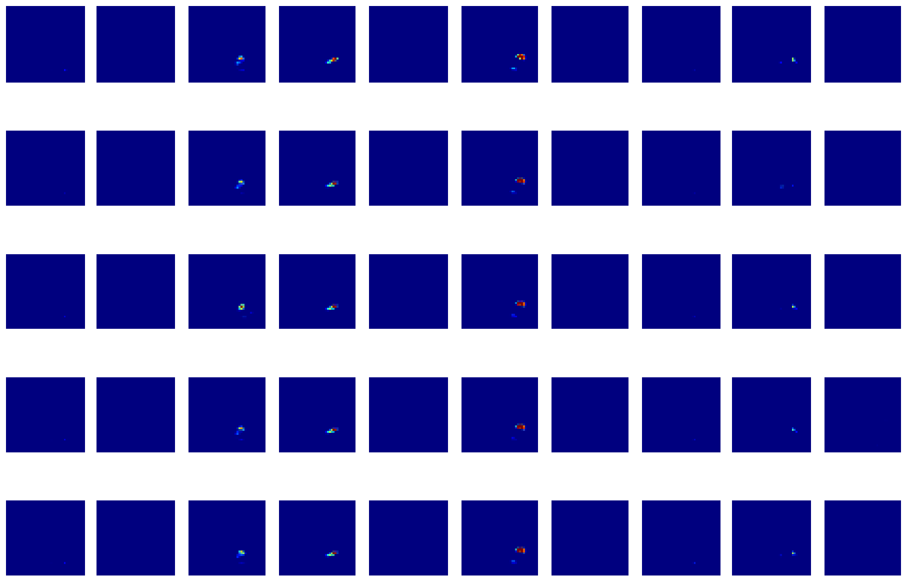
0



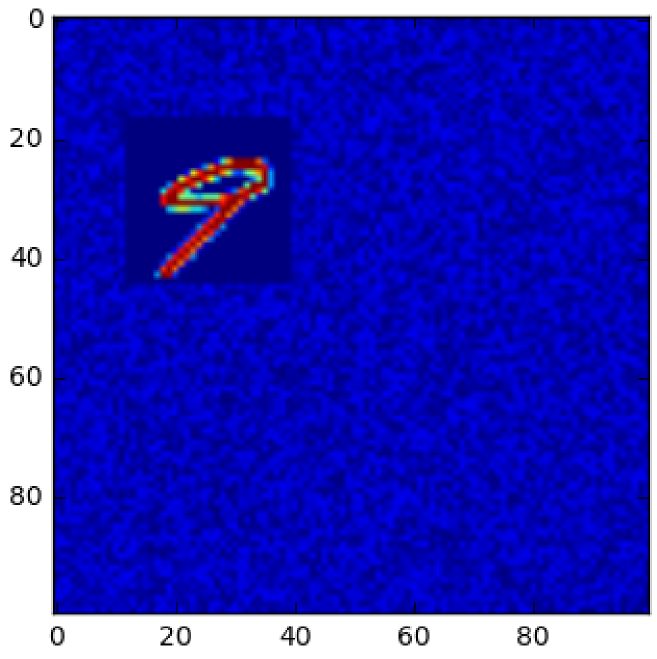


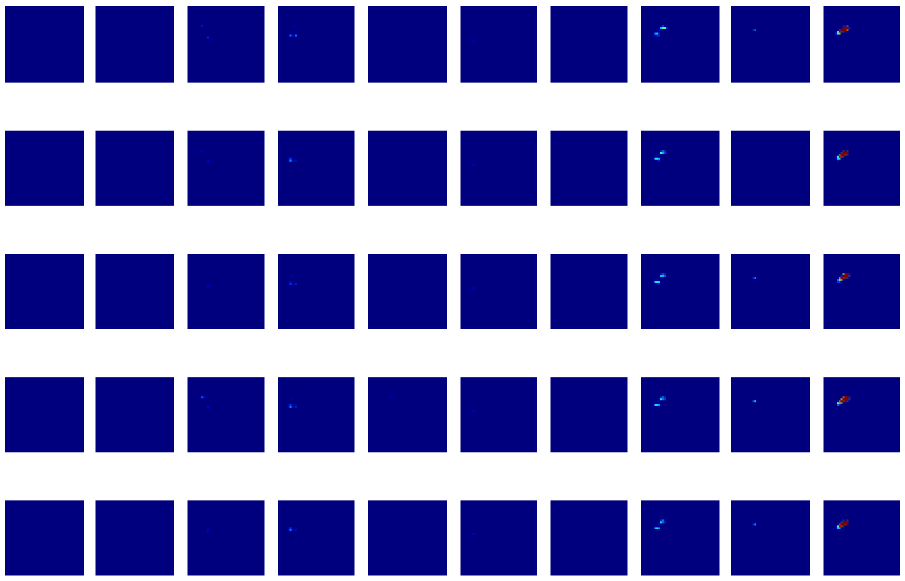
5





9





```
In [ ]:
```