show_results

November 3, 2016

1 Load model

1.1 Model

conv1_1 = new_conv_layer(image, [9, 9, 1 , 16], “conv1_1”) conv1_2 = new_conv_layer(conv1_1, [9, 9, 16, 16], “conv1_2”)
conv2_1 = new_conv_layer(conv1_2, [9, 9, 16, 16], “conv2_1”) conv2_2 = new_conv_layer(conv2_1, [9, 9, 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

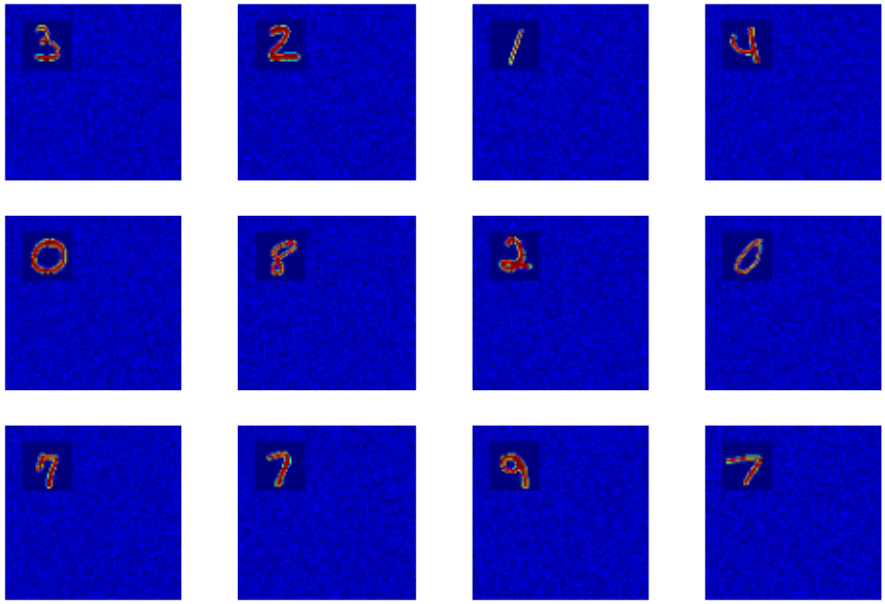
L2 on weights (5e-5) L1 on GAP

In [8]:
```
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 [9]:
```
"""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 [10]: """Perform training"""
         gen = training_generator(lr=lr, lr_decay=lr_decay, back_size=back_size, no
         for _ in range(25):
             _,accs = gen.next()
             print "max acc so far : "+str(max(accs)*100)
```

```
***** EPOCH 0 ******
1-Loss on testset is 1.783015
1-Accuracy now is 38.86
2-Loss on testset is 1.780118
2-Accuracy now is 38.94
lr now is 0.00450
max acc so far : 38.86

***** EPOCH 1 ******
1-Loss on testset is 0.806614
1-Accuracy now is 86.04
2-Loss on testset is 0.782621
2-Accuracy now is 86.87
lr now is 0.00405
```

```
max acc so far : 86.04

***** EPOCH 2 ******
1-Loss on testset is 0.394757
1-Accuracy now is 96.23
2-Loss on testset is 0.382632
2-Accuracy now is 96.52
lr now is 0.00365
max acc so far : 96.23

***** EPOCH 3 ******
1-Loss on testset is 0.491537
1-Accuracy now is 91.09
2-Loss on testset is 0.457808
2-Accuracy now is 92.35
lr now is 0.00328
max acc so far : 96.23

***** EPOCH 4 ******
1-Loss on testset is 0.345862
1-Accuracy now is 96.73
2-Loss on testset is 0.341597
2-Accuracy now is 96.85
lr now is 0.00295
max acc so far : 96.73

***** EPOCH 5 ******
1-Loss on testset is 0.262977
1-Accuracy now is 98.19
2-Loss on testset is 0.246605
2-Accuracy now is 98.48
lr now is 0.00266
max acc so far : 98.19

***** EPOCH 6 ******
1-Loss on testset is 0.358894
1-Accuracy now is 95.41
2-Loss on testset is 0.354807
2-Accuracy now is 95.36
lr now is 0.00239
max acc so far : 98.19

***** EPOCH 7 ******
1-Loss on testset is 1.783721
1-Accuracy now is 58.06
2-Loss on testset is 0.303575
2-Accuracy now is 96.48
lr now is 0.00215
```

```
max acc so far : 98.19

***** EPOCH 8 ******
1-Loss on testset is 2.312777
1-Accuracy now is 36.61
2-Loss on testset is 0.258305
2-Accuracy now is 97.75
lr now is 0.00194
max acc so far : 98.19

***** EPOCH 9 ******
1-Loss on testset is 0.258836
1-Accuracy now is 97.95
2-Loss on testset is 0.221195
2-Accuracy now is 98.53
lr now is 0.00174
max acc so far : 98.19

***** EPOCH 10 ******
1-Loss on testset is 1.087707
1-Accuracy now is 69.45
2-Loss on testset is 0.253745
2-Accuracy now is 97.61
lr now is 0.00157
max acc so far : 98.19

***** EPOCH 11 ******
1-Loss on testset is 0.378035
1-Accuracy now is 95.61
2-Loss on testset is 0.203262
2-Accuracy now is 98.88
lr now is 0.00141
max acc so far : 98.19

***** EPOCH 12 ******
1-Loss on testset is 0.611179
1-Accuracy now is 87.06
2-Loss on testset is 0.188675
2-Accuracy now is 99.15
lr now is 0.00127
max acc so far : 98.19

***** EPOCH 13 ******
1-Loss on testset is 2.695050
1-Accuracy now is 24.31
2-Loss on testset is 0.190207
2-Accuracy now is 99.06
lr now is 0.00114
```

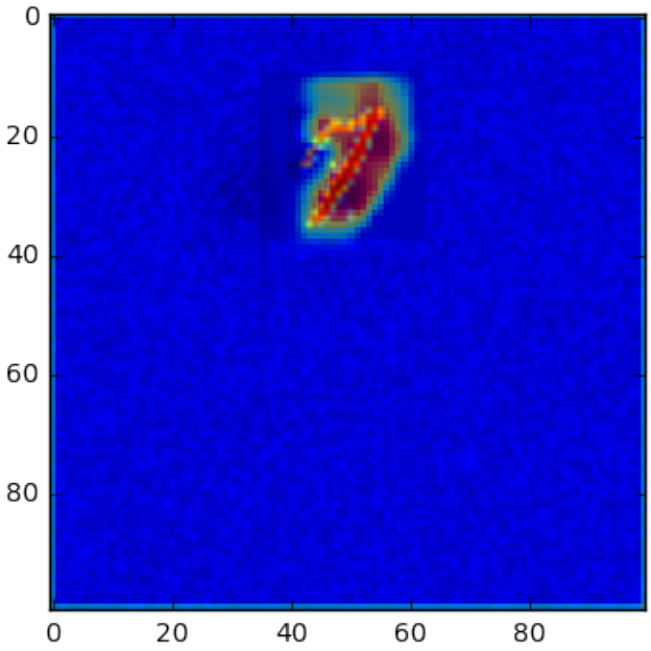
```
max acc so far : 98.19

***** EPOCH 14 ******
1-Loss on testset is 7.453392
1-Accuracy now is 9.90
2-Loss on testset is 0.216337
2-Accuracy now is 98.57
lr now is 0.00103
max acc so far : 98.19

***** EPOCH 15 ******
1-Loss on testset is 1.693115
1-Accuracy now is 40.93
2-Loss on testset is 0.186272
2-Accuracy now is 99.17
lr now is 0.00093
max acc so far : 98.19

***** EPOCH 16 ******
1-Loss on testset is 3.392904
1-Accuracy now is 13.81
2-Loss on testset is 0.190582
2-Accuracy now is 99.00
lr now is 0.00083
max acc so far : 98.19

***** EPOCH 17 ******
1-Loss on testset is 4.739895
1-Accuracy now is 24.01
2-Loss on testset is 0.182348
2-Accuracy now is 99.07
lr now is 0.00075
max acc so far : 98.19

***** EPOCH 18 ******
1-Loss on testset is 8.819411
1-Accuracy now is 19.38
2-Loss on testset is 0.176615
2-Accuracy now is 99.19
lr now is 0.00068
max acc so far : 98.19

***** EPOCH 19 ******
1-Loss on testset is 14.605368
1-Accuracy now is 15.39
2-Loss on testset is 0.177855
2-Accuracy now is 99.30
lr now is 0.00061
```

```
max acc so far : 98.19

***** EPOCH 20 ******
1-Loss on testset is 17.154383
1-Accuracy now is 9.73
2-Loss on testset is 0.182276
2-Accuracy now is 99.11
lr now is 0.00055
max acc so far : 98.19

***** EPOCH 21 ******
1-Loss on testset is 25.296286
1-Accuracy now is 9.58
2-Loss on testset is 0.177342
2-Accuracy now is 99.31
lr now is 0.00049
max acc so far : 98.19

***** EPOCH 22 ******
1-Loss on testset is 25.002905
1-Accuracy now is 9.59
2-Loss on testset is 0.179805
2-Accuracy now is 99.20
lr now is 0.00044
max acc so far : 98.19

***** EPOCH 23 ******
1-Loss on testset is 26.465839
1-Accuracy now is 9.58
2-Loss on testset is 0.178272
2-Accuracy now is 99.14
lr now is 0.00040
max acc so far : 98.19

***** EPOCH 24 ******
1-Loss on testset is 33.614231
1-Accuracy now is 9.58
2-Loss on testset is 0.174226
2-Accuracy now is 99.24
lr now is 0.00036
max acc so far : 98.19
```

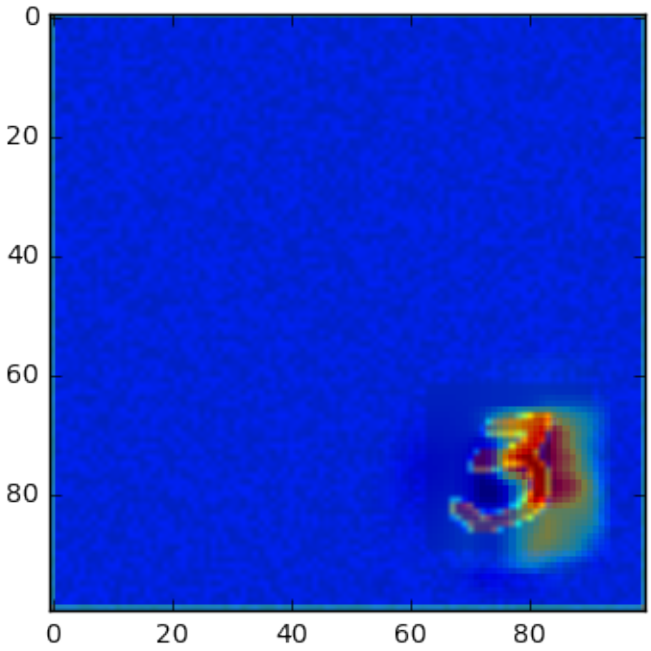
```
In [11]: back_size  = 100

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

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

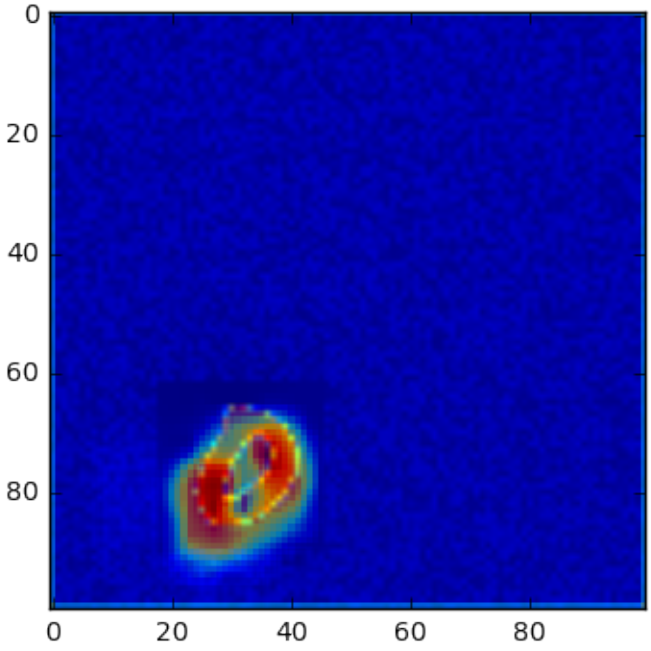
```
--------------------------------------------------------------------------------
prediction is : 7 with 5.761
```



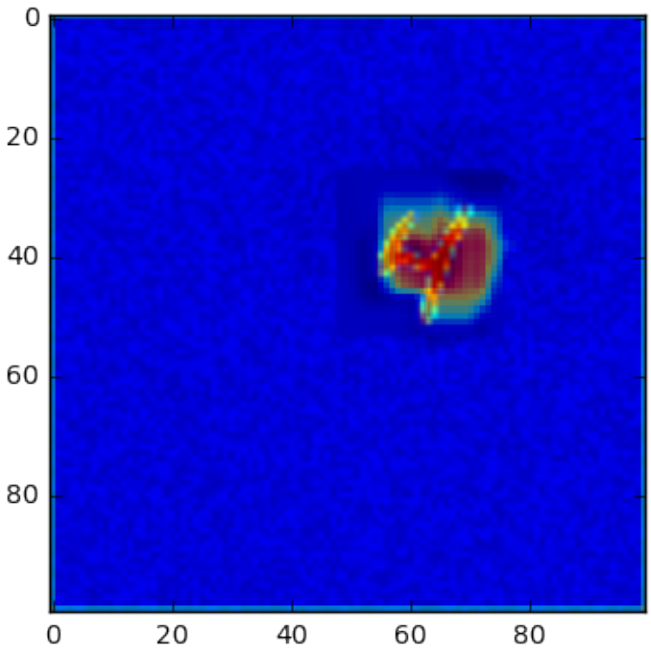

```
--------------------------------------------------------------------------------
prediction is : 3 with 8.994
```



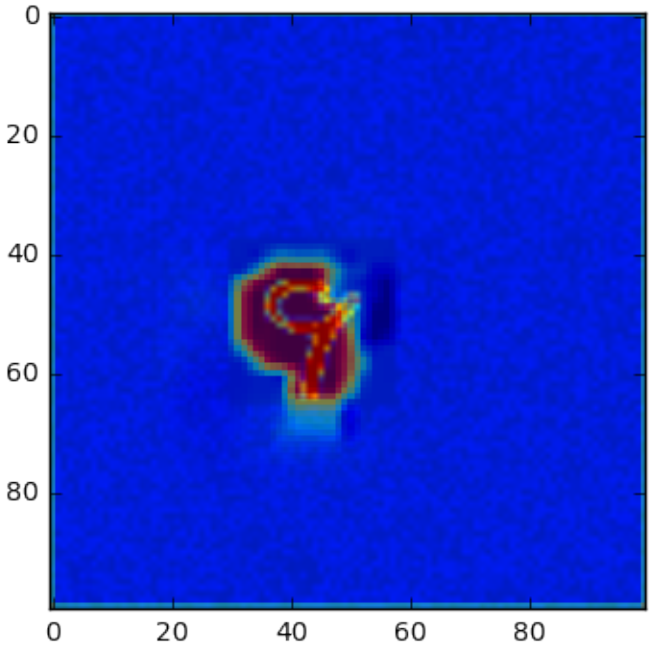
```
prediction is : 0 with 8.164
```



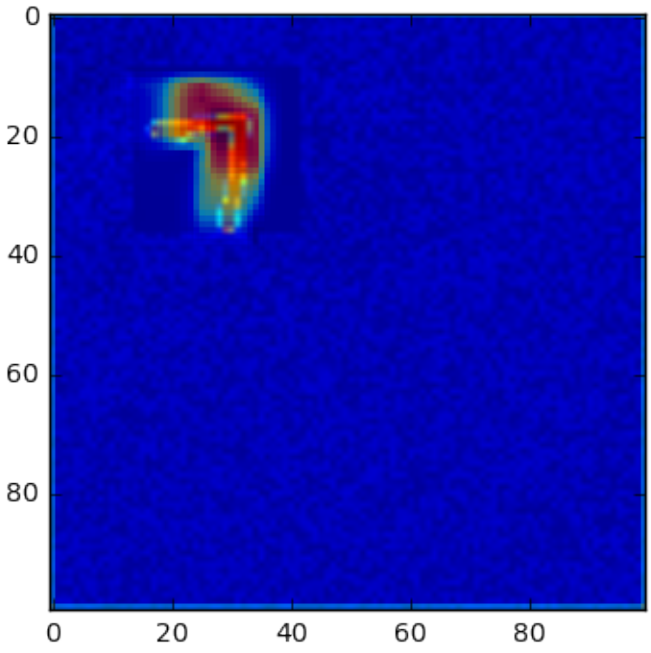
```
--------------------------------------------------------------------------------
prediction is : 4 with 7.430
```



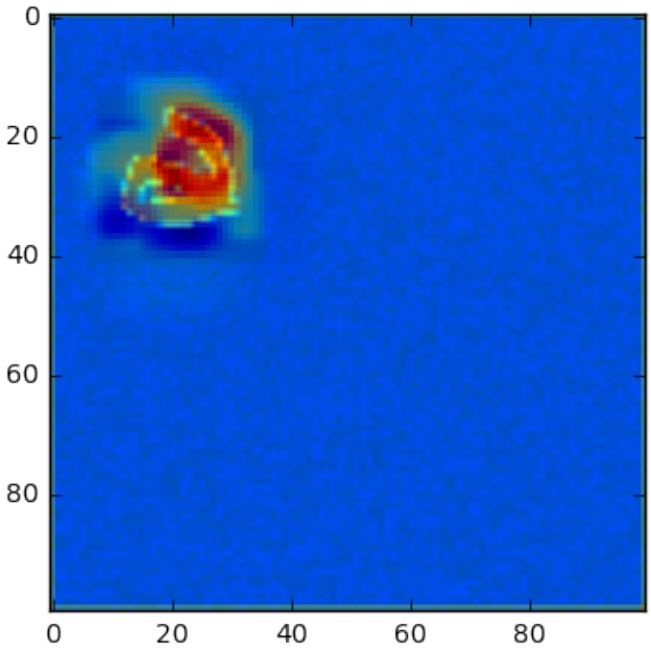

```
--------------------------------------------------------------------------------
prediction is : 9 with 3.864
```



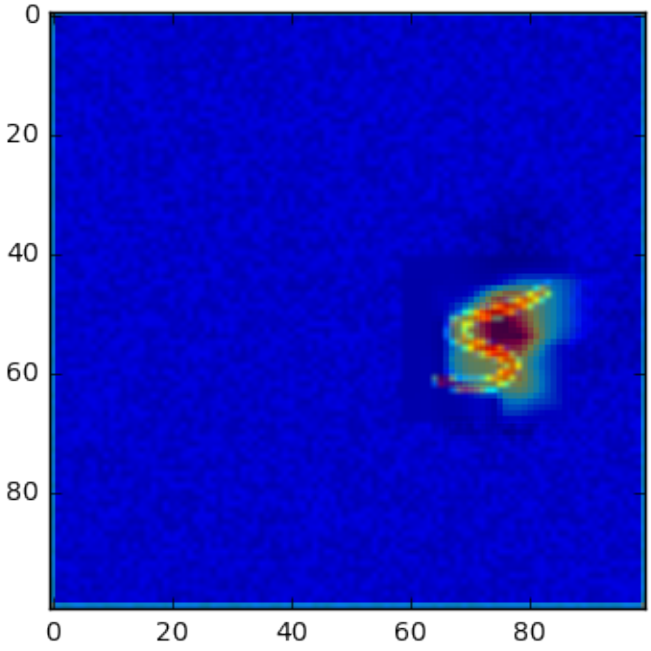
```
--------------------------------------------------------------------------------
prediction is : 7 with 9.307
```



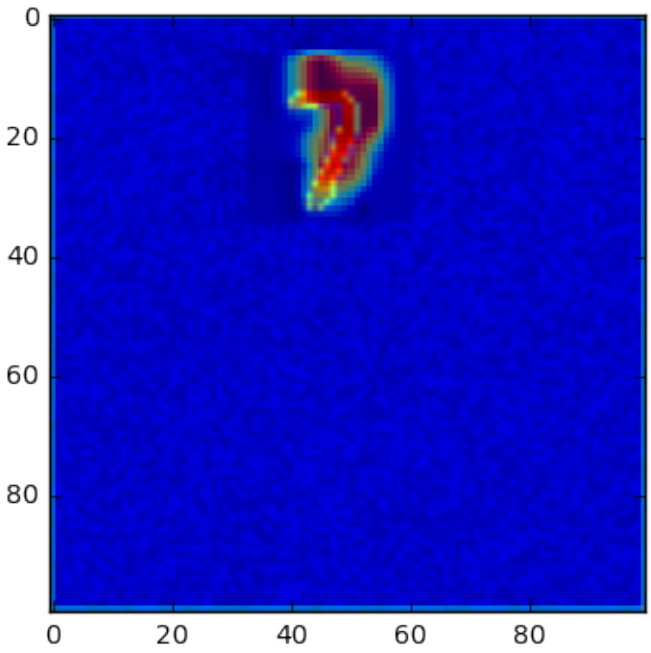

```
--------------------------------------------------------------------------------
prediction is : 2 with 7.757
```



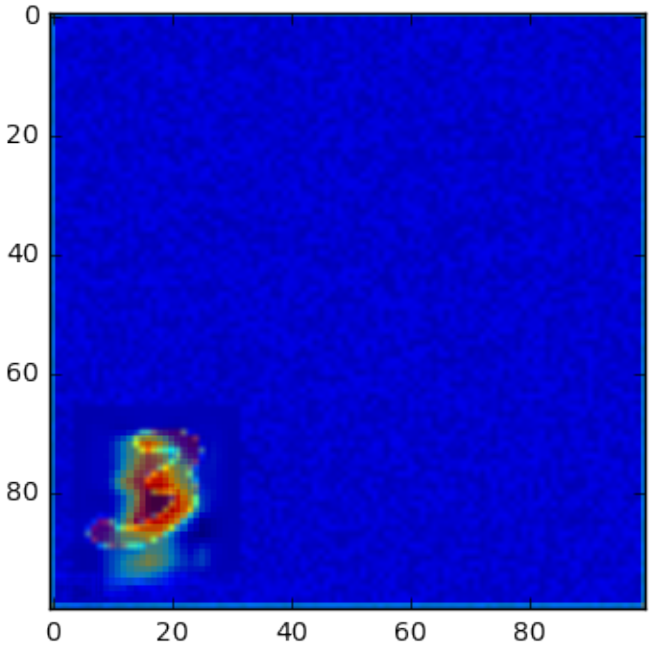
```
prediction is : 5 with 9.612
```



```
--------------------------------------------------------------------------------
prediction is : 7 with 7.440
```




```
--------------------------------------------------------------------------------
prediction is : 3 with 9.089
```



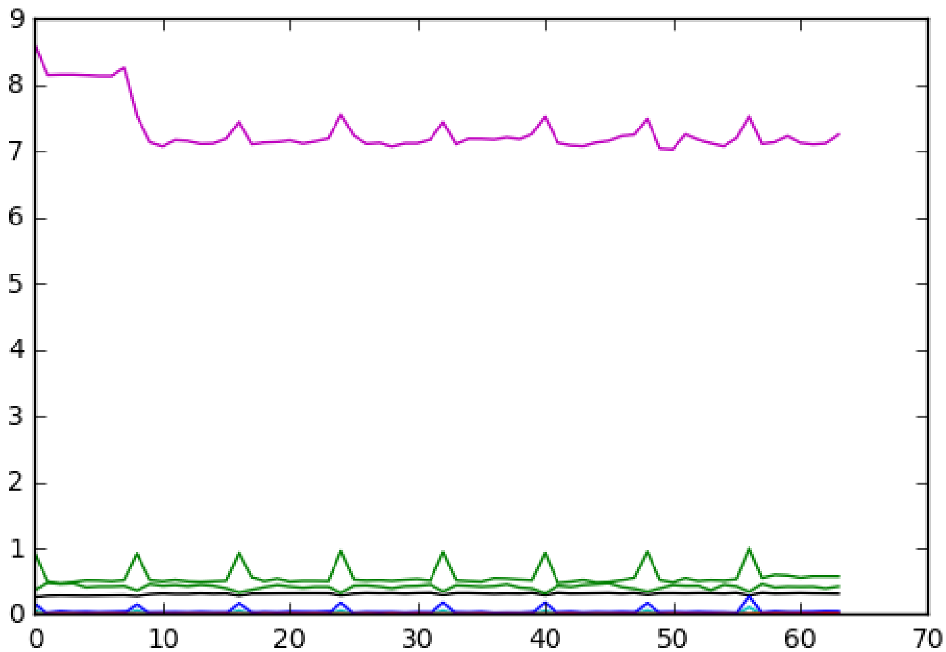
```
In [12]: accuracies = []
         for _ in range(100):
             digit,lbl  = utils.get_batch('test', 1, .1).next()
             digit      = digit[0].reshape(28,28)
             img        = np.random.random((100,100))*.1
             height     = img.shape[0]
             width      = img.shape[1]
             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_m
             pred  = np.argmax(preds[-1])

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

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

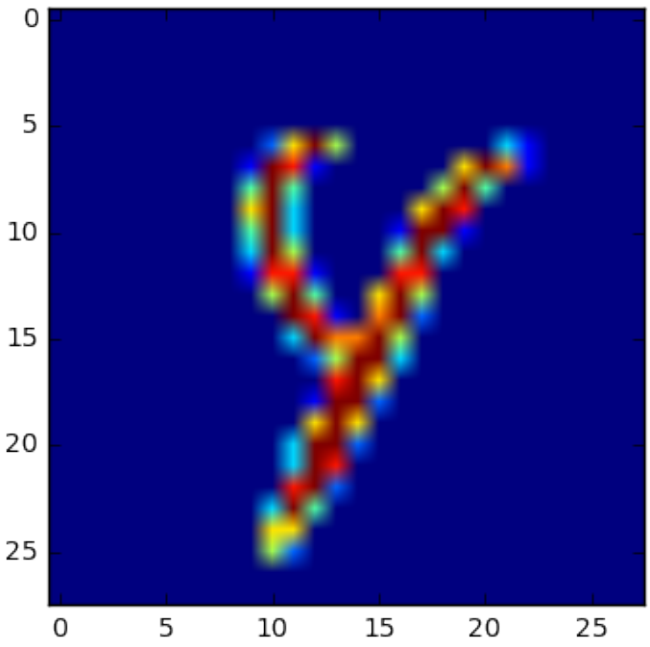
```
0.9890625
```

```
In [13]: 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()
```

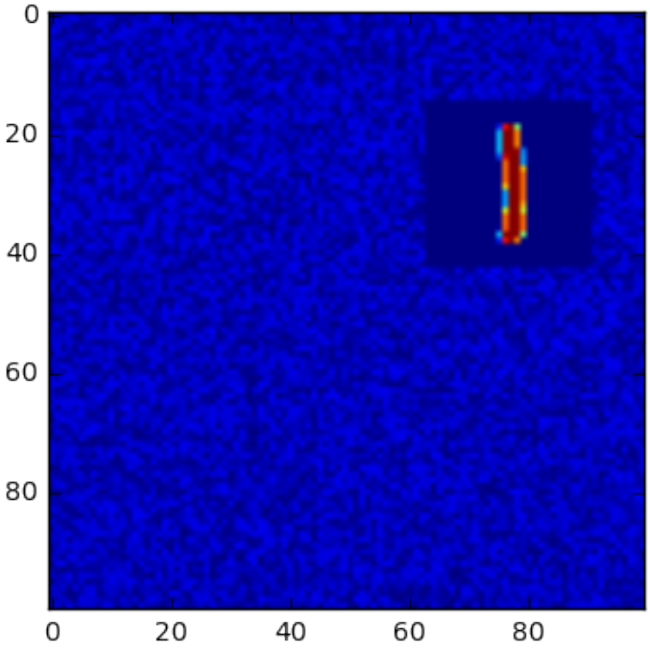
```
[4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4]
100.0
```

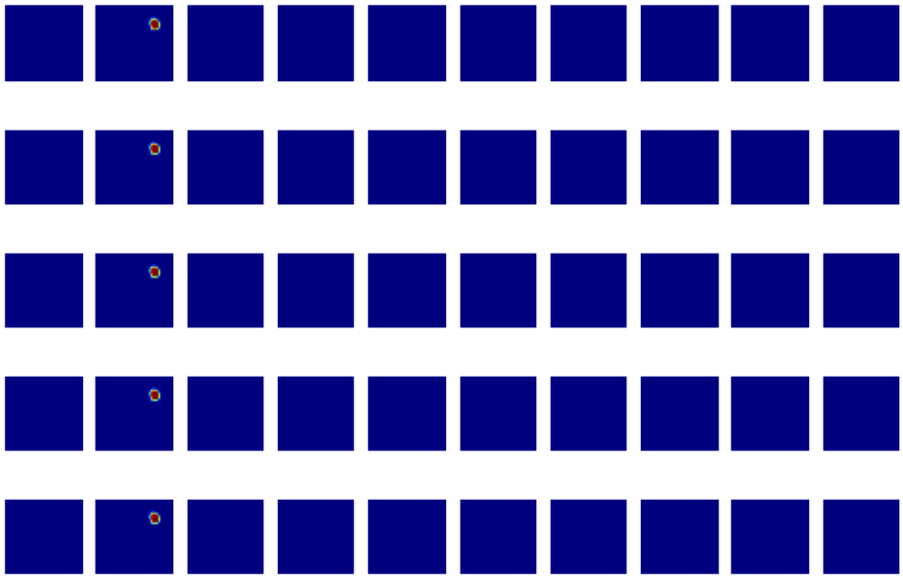




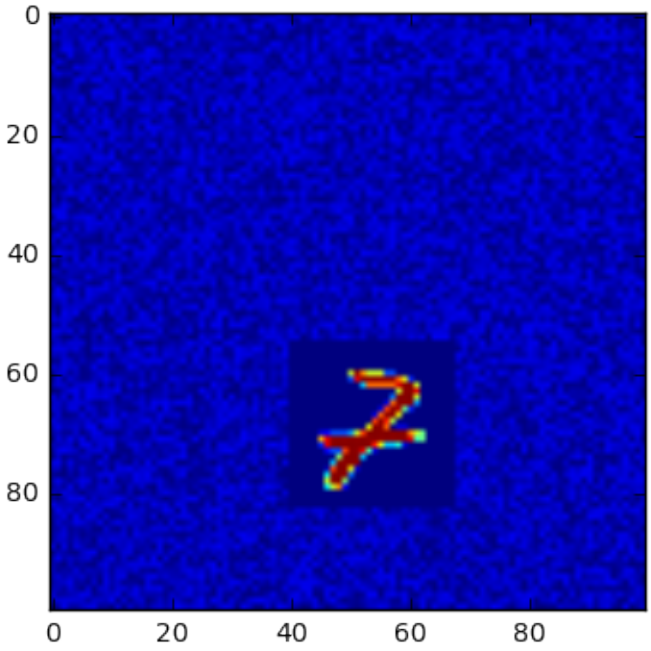

```
In [14]: for _ in range(10):
             simple_model.plot_classes_maps(100, n_CCN)
```

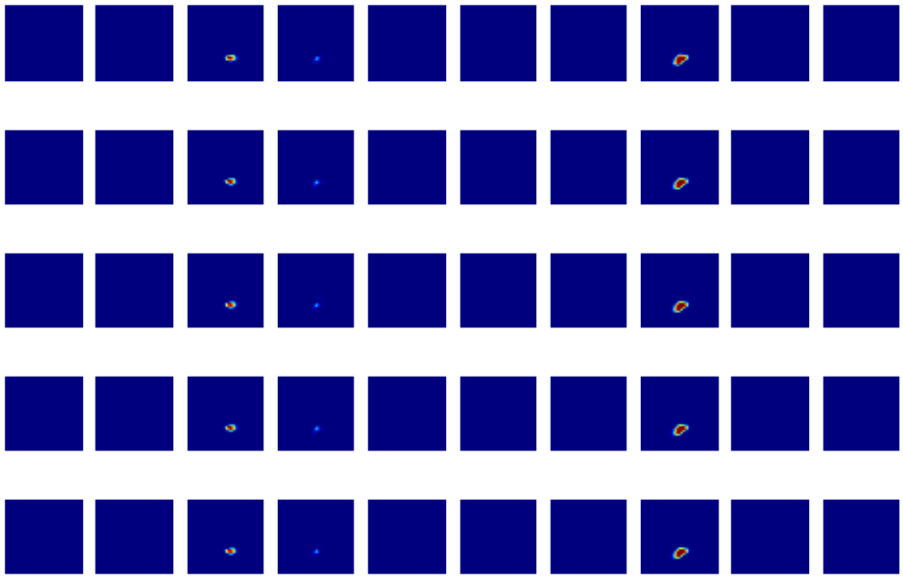
```
1
```



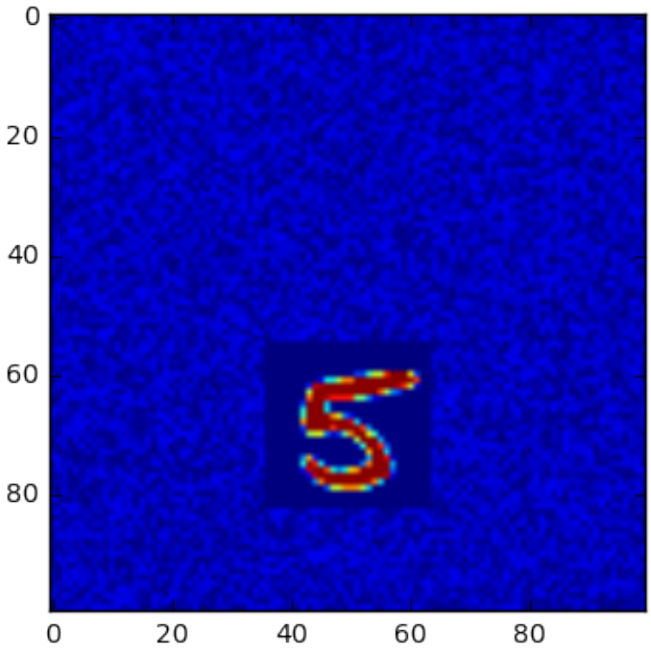


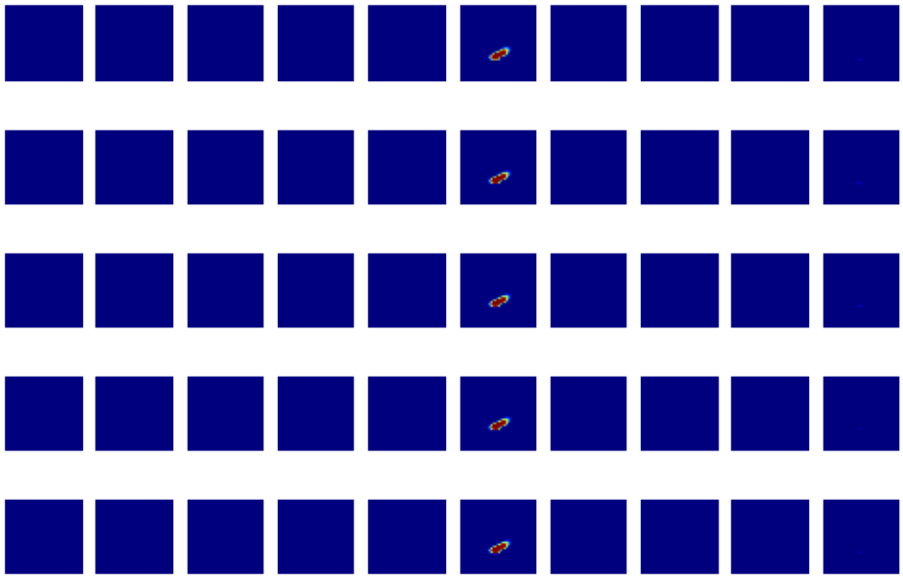
7



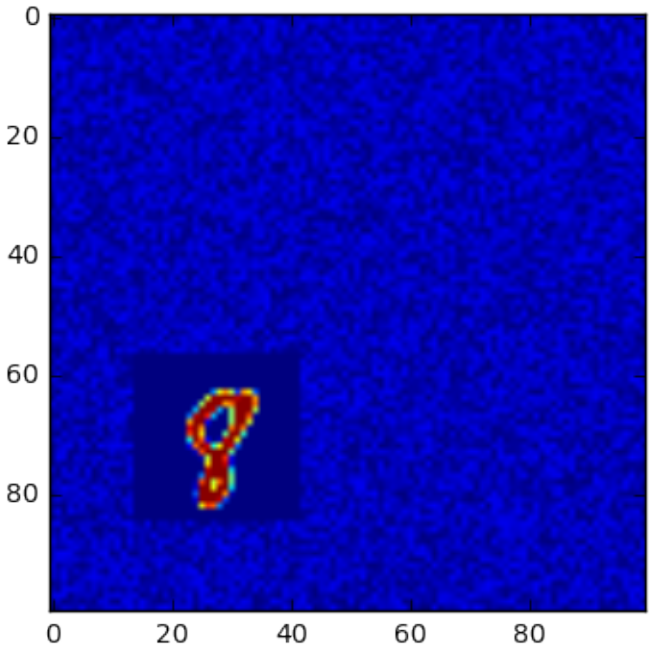


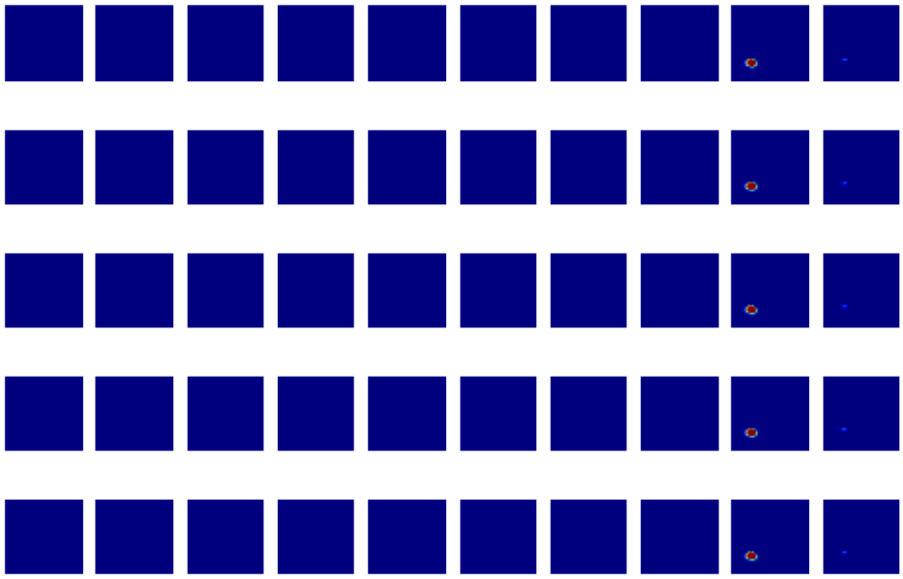
5



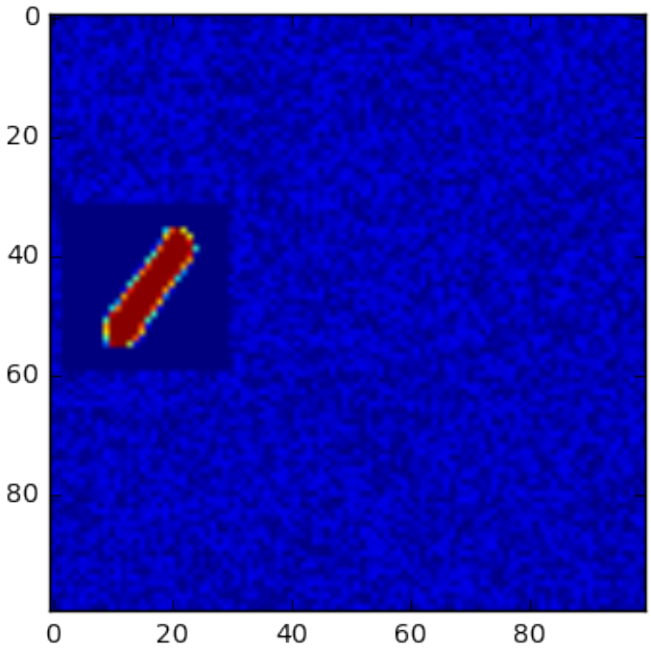


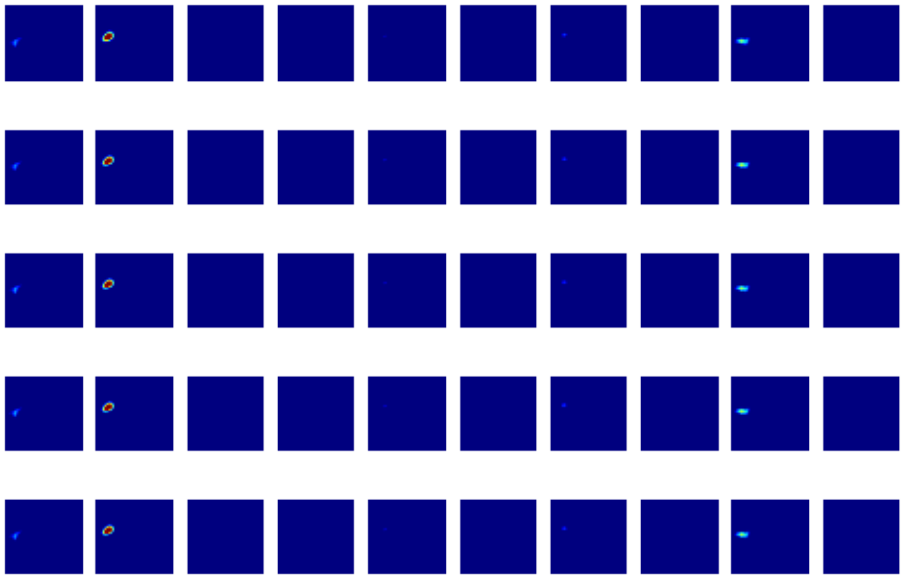
8



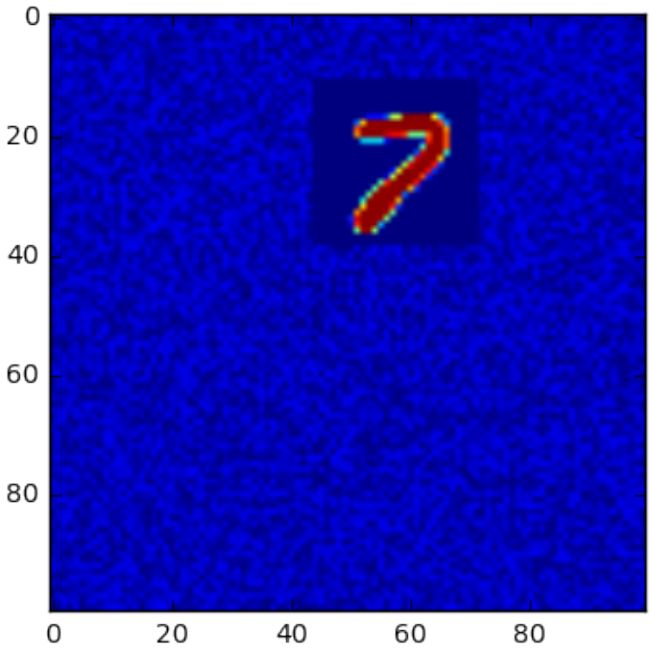


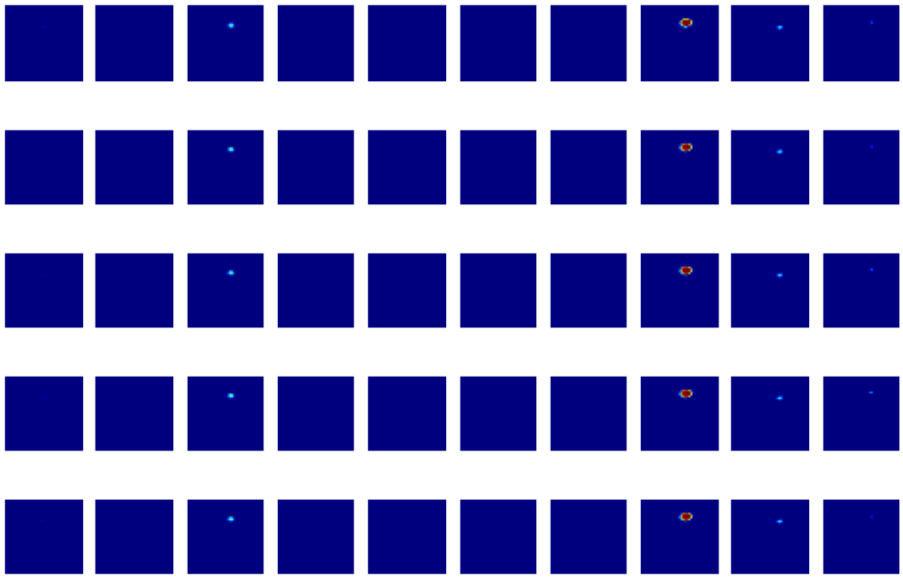
1



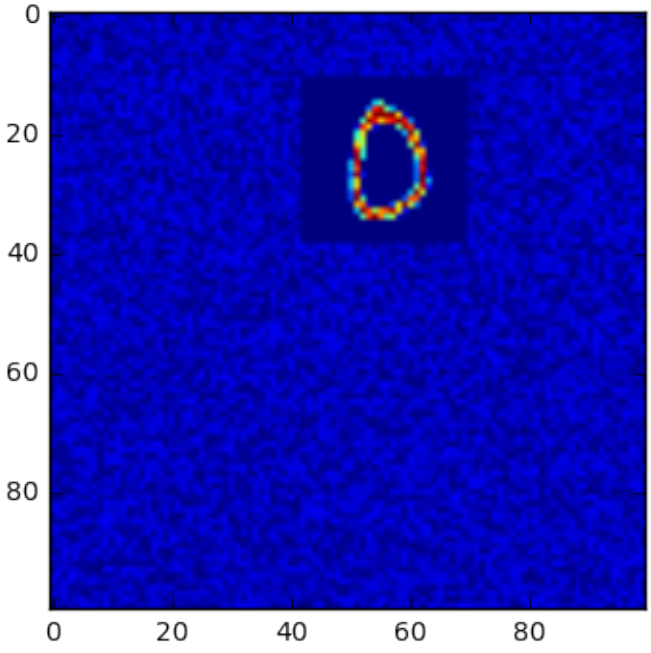


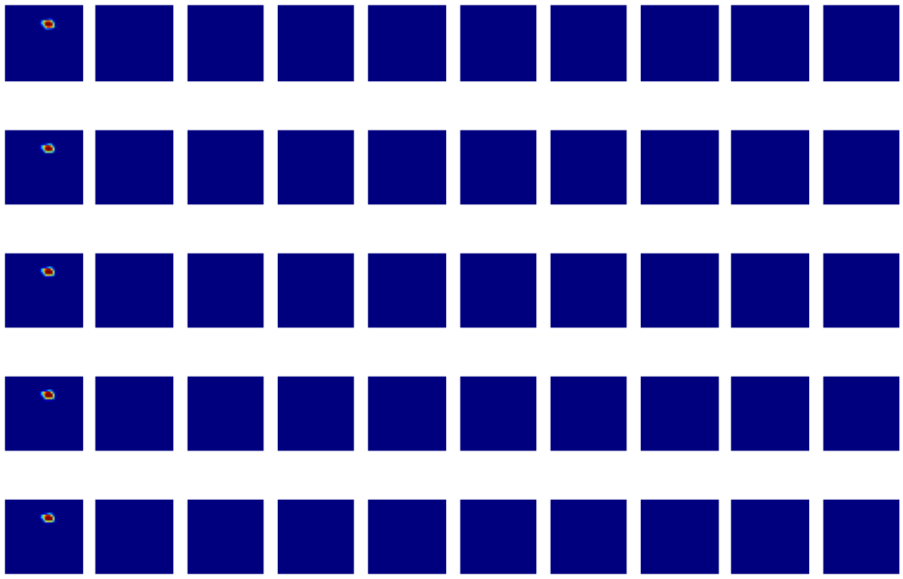
7



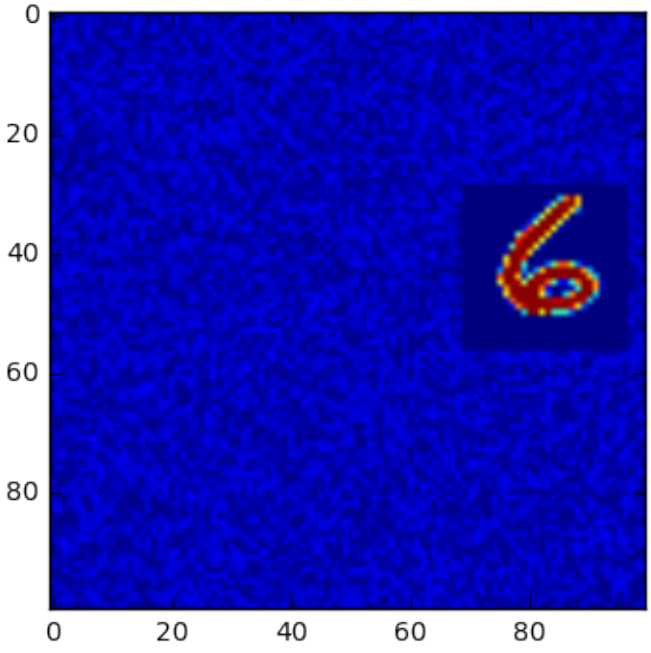


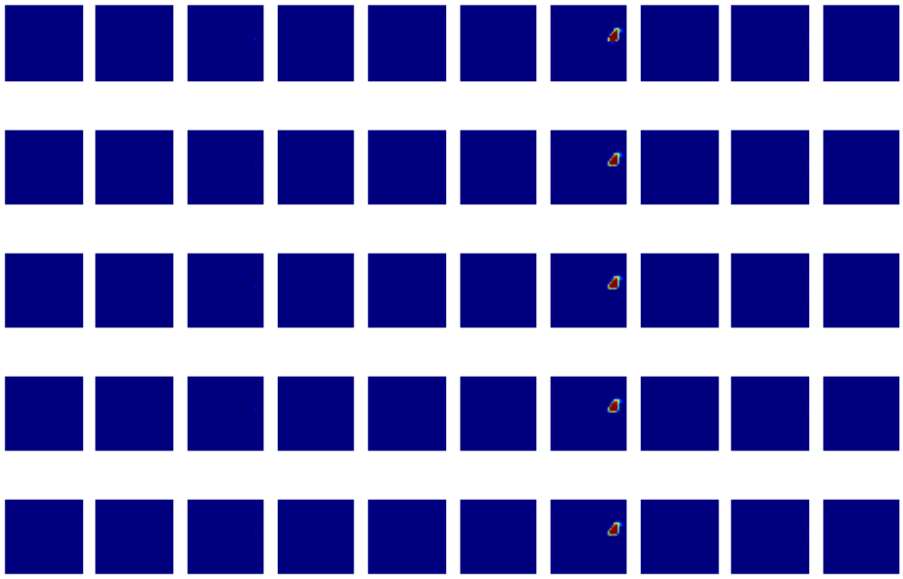
0



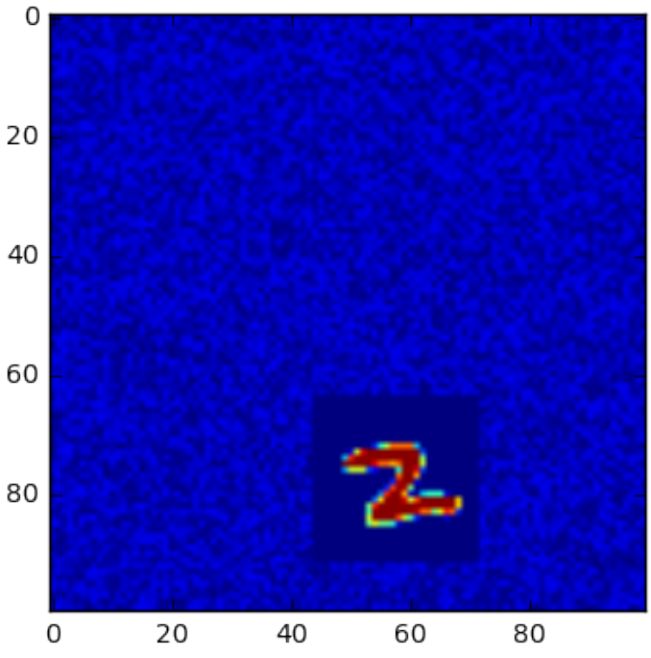


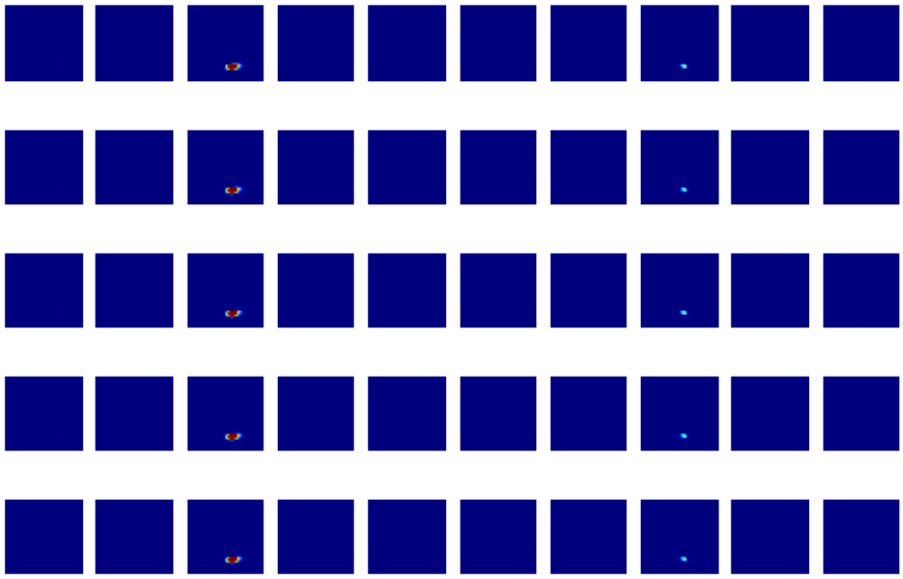
6



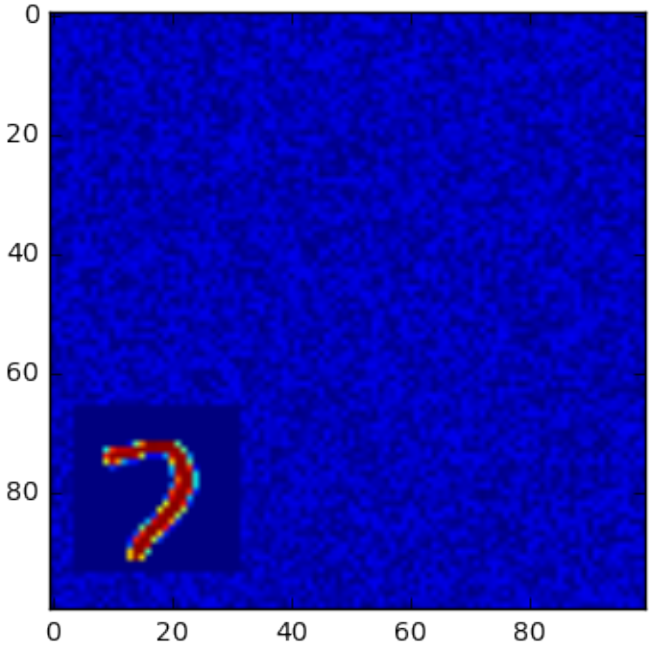


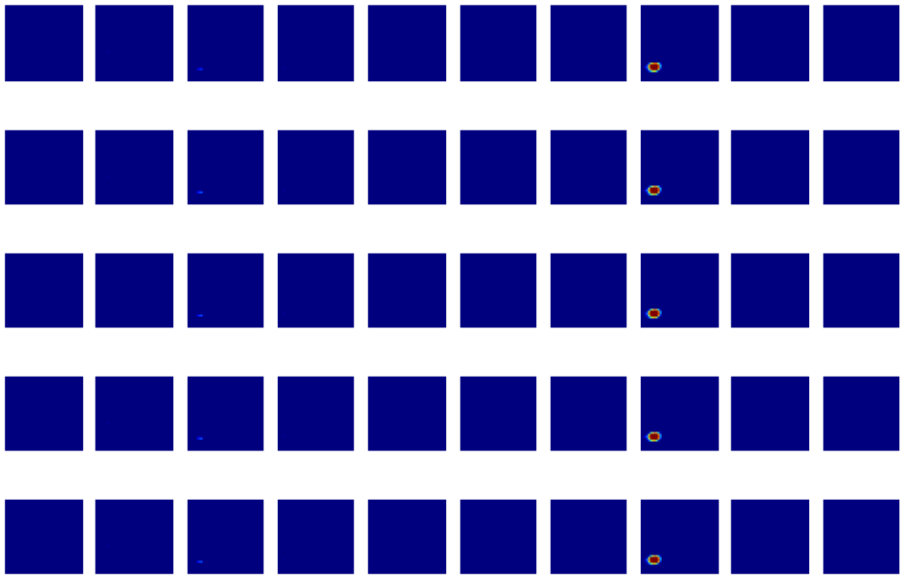
2





7





In [ ]: