Lab 3 Report: Inference Attacks on Deep Neural Networks

Sui Huang sh4507@nyu.edu

1. Methodology used to implement the AM algorithm:

I used gradient decent mythology. The code realized in function named 'image_optimizer'. Function image_optimizer(image_old, i, lr, print_=True) takes 4 arguments:

image_old is the initial image I generated using either random or mean value, and this image is generated using function named 'generate_x'.

i: is the target label we want the initial picture to be updated. For example, i=5, then the target label will be [0,0,0,0,0,1,0,0,0,0]

lr: Learing rate

print_: whether I want to print the result(disabled in Q4 because it's too much)

Code for optimizing gradient and update the image:

if (prob_new[:,i]-prob_old[:,i]>= 1e-8): (stops when the probability does not increase any more)

gradient = tf.gradients(xs = x, ys = cost) (compute gradient)

image_new = tf.clip_by_value(x - tf.fill([784], step_size)/gradient,0,1) *(image = image – lr/gradient, clip from 0 to 1 because I don't want the value exceed the image range)*

image_new, gradient = sess.run([image_new , gradient], feed_dict={x: image, y: generate_y(j).reshape(1,10)}) (session run)

image_new = image_new[0,:,:] (decrease one dimension)

image = image_new (update image to calculate gradients)

I did not use adaptive learning rate. But used different learning rate (as step size) for different initial images.

PART 1: 1e-1, PART 2：1e-2, PART 3&Q4：1e-5

2. **Recovered images for each class (10 per problem) and the computed RMSE for that image. Also indicate the final error.**

PART 1:

Image recovered for each label:

RMSE for each recovered image:
['0.63', '0.58', '0.58', '0.59', '0.58', '0.66', '0.58', '0.59', '0.57', '0.58']

Final error: 0.593434440239

PART 2:

Initial image using all mean:

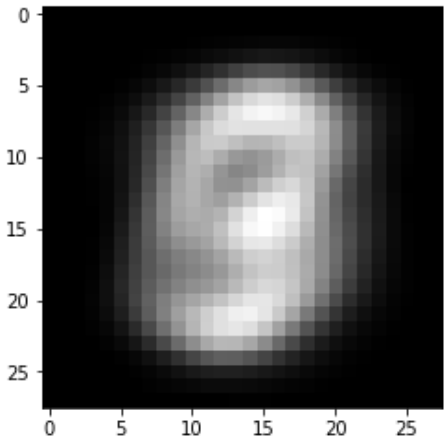
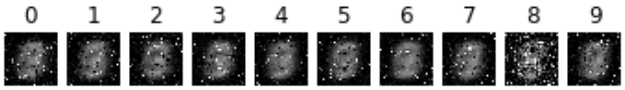


Image recovered for each label:



RMSE for each recovered image:
['0.24', '0.23', '0.23', '0.22', '0.22', '0.23', '0.23', '0.22', '0.31', '0.23']

Final error: 0.237067775917

PART 3:

```
Initial image for each class:
```


```
Image recovered for each label:
```


```
RMSE for each recovered image:
['0.31', '0.13', '0.19', '0.17', '0.17', '0.18', '0.18', '0.15', '0.19', '0.16']

Final error:  0.183498833561
```

3. **Tabulate the prediction accuracy versus the final error for each value of λ. Also indicate the value of λ used.**

Alpha	Accuracy	RMSE	W1 Lambda	W3 Lambda
0.01	0.914667	0.184212	0.01	0.000270
0.10	0.905333	0.178094	0.10	0.002703
0.50	0.452000	0.175666	0.50	0.013517
1.00	0.234667	0.192974	1.00	0.027035

4. **Python code along with any instructions required to execute the code.**
   All codes and results can be reproduced using the note book named:' Inference_ Attacks_on_Deep_Neural Networks'

5. **Some thoughts**
   I think I did not do very well in part4. And the result is that adding noise can not increase RMSE in a very large level. Maybe I did the DP wrong. It's sad because I don't have more time.