

# Adversarial Robustness-toolbox

## Generate Counter Examples

Fast Gradient Method, Basic Iterative Method, Deep Fool

November 9, 2023

### 1. Label Counts

Basic Information:

Total number of images: 70000

Size of each image: (28, 28)

Number of unique labels: 10

Distribution of Labels:

Label 0: 6903

Label 1: 7877

Label 2: 6990

Label 3: 7141

Label 4: 6824

Label 5: 6313

Label 6: 6876

Label 7: 7293

Label 8: 6825

Label 9: 6958

Pixel Intensity Statistics:

Minimum pixel value: 0

Maximum pixel value: 255

Mean pixel value: 33.39

Standard deviation of pixel values: 78.65

Balance Check:

The dataset is not balanced across different labels.

Figure 1: Basic Information of Dataset

---

**Algorithm 1** Evaluation of a Pre-trained Model on MNIST Dataset

---

**Require:** Pre-trained model path  $\mathcal{P}_{\text{model}}$ , MNIST dataset

**Ensure:** Test accuracy, Correct examples and labels saved

```

1: function LOADMODEL( $\mathcal{P}_{\text{model}}$ )
2:    $\mathcal{M} \leftarrow$  load model from  $\mathcal{P}_{\text{model}}$ 
3:   return  $\mathcal{M}$ 
4: function LOADMNISTDATA
5:    $(\mathcal{X}_{\text{train}}, \mathcal{Y}_{\text{train}}), (\mathcal{X}_{\text{test}}, \mathcal{Y}_{\text{test}}) \leftarrow$  MNIST data
6:   return  $(\mathcal{X}_{\text{train}}, \mathcal{Y}_{\text{train}}), (\mathcal{X}_{\text{test}}, \mathcal{Y}_{\text{test}})$ 
7: function PREPROCESSDATA( $\mathcal{X}$ )
8:    $\mathcal{X} \leftarrow$  reshape and normalize  $\mathcal{X}$ 
9:   return  $\mathcal{X}$ 
10: function ONEHOTENCODE( $\mathcal{Y}$ )
11:    $\mathcal{Y}_{\text{encoded}} \leftarrow$  one-hot encode  $\mathcal{Y}$ 
12:   return  $\mathcal{Y}_{\text{encoded}}$ 
13: function EVALUATEMODEL( $\mathcal{M}, \mathcal{X}_{\text{test}}, \mathcal{Y}_{\text{test}}$ )
14:    $\mathcal{P} \leftarrow$  model predictions for  $\mathcal{X}_{\text{test}}$ 
15:    $\mathcal{L}, \mathcal{A} \leftarrow$  evaluate  $\mathcal{M}$  on  $\mathcal{X}_{\text{test}}$  and  $\mathcal{Y}_{\text{test}}$ 
16:   return  $\mathcal{P}, \mathcal{L}, \mathcal{A}$ 
17: function SAVECORRECT( $\mathcal{X}_{\text{test}}, \mathcal{Y}_{\text{test}}, \mathcal{P}$ )
18:    $\mathcal{I}_{\text{correct}} \leftarrow$  indices where  $\mathcal{P}$  equals  $\mathcal{Y}_{\text{test}}$ 
19:    $\mathcal{X}_{\text{correct}} \leftarrow$  examples from  $\mathcal{X}_{\text{test}}$  at  $\mathcal{I}_{\text{correct}}$ 
20:    $\mathcal{Y}_{\text{correct}} \leftarrow$  labels from  $\mathcal{Y}_{\text{test}}$  at  $\mathcal{I}_{\text{correct}}$ 
21:   Save  $\mathcal{X}_{\text{correct}}$  and  $\mathcal{Y}_{\text{correct}}$  to disk
22:  $\mathcal{M} \leftarrow$  LOADMODEL( $\mathcal{P}_{\text{model}}$ )
23:  $(\mathcal{X}_{\text{train}}, \mathcal{Y}_{\text{train}}), (\mathcal{X}_{\text{test}}, \mathcal{Y}_{\text{test}}) \leftarrow$  LOADMNISTDATA
24:  $\mathcal{X}_{\text{train}} \leftarrow$  PREPROCESSDATA( $\mathcal{X}_{\text{train}}$ )
25:  $\mathcal{X}_{\text{test}} \leftarrow$  PREPROCESSDATA( $\mathcal{X}_{\text{test}}$ )
26:  $\mathcal{Y}_{\text{train}} \leftarrow$  ONEHOTENCODE( $\mathcal{Y}_{\text{train}}$ )
27:  $\mathcal{Y}_{\text{test}} \leftarrow$  ONEHOTENCODE( $\mathcal{Y}_{\text{test}}$ )
28:  $\mathcal{P}, \mathcal{L}, \mathcal{A} \leftarrow$  EVALUATEMODEL( $\mathcal{M}, \mathcal{X}_{\text{test}}, \mathcal{Y}_{\text{test}}$ )
29: Print  $\mathcal{A}$  as test accuracy
30: SAVECORRECT( $\mathcal{X}_{\text{test}}, \mathcal{Y}_{\text{test}}, \mathcal{P}$ )

```

---

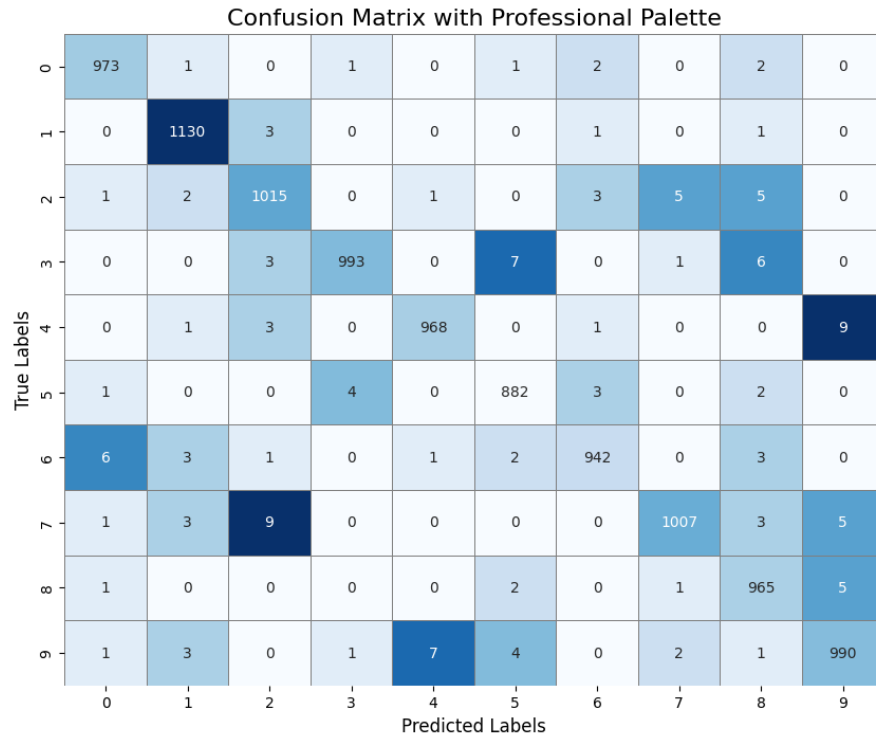


Figure 2: CM FSGM

Table 1: Label counts

	Total MNIST	Training Set	Test Set	Predict Labels	Correct Labels
0	6903	5923	980	992	973
1	7877	6742	1135	1157	1130
2	6990	5958	1032	1037	1015
3	7141	6131	1010	999	993
4	6824	5842	982	983	968
5	6313	5421	892	904	882
6	6876	5918	958	943	942
7	7293	6265	1028	1024	1007
8	6825	5851	974	954	965
9	6958	5949	1009	1007	990
Total	70000	60000	10000	10000	9865

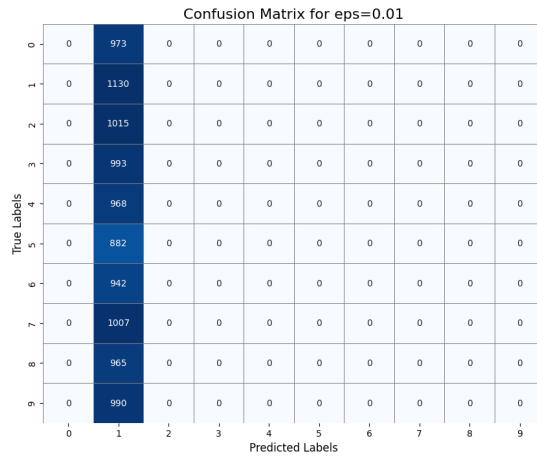
## 2. FSGM without corrected variables

### Algorithm 2 Adversarial Example Generation and Analysis

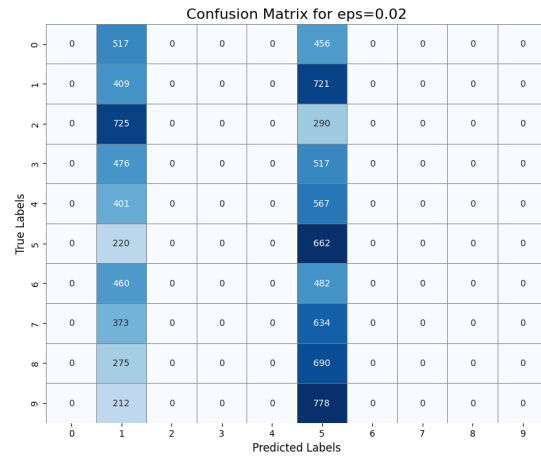
```

1: Input: classifier, correct_examples, correct_labels
2: Output: results_df, confusion matrices
3: eps_range  $\leftarrow [0.01, 0.02, \dots, 0.6]$ 
4: results_df  $\leftarrow$  DataFrame with columns ['eps', 'total_correct', 'total_adv', 'correct_adv_counts']
5: for eps  $\in$  eps_range do
6:   attack  $\leftarrow$  FastGradientMethod(classifier, eps)
7:   x_adv  $\leftarrow$  attack.generate(x=correct_examples)
8:   y_adv  $\leftarrow$  argmax(classifier.predict(x_adv), axis = 1)
9:   adv_counts  $\leftarrow$  count_unique(y_adv)
10:  cm  $\leftarrow$  confusion_matrix(correct_labels, y_adv)
11:  Save heatmap of cm to file with filename based on eps
12:  results_df.append({'eps' : eps, 'total_correct' : length(correct_labels), 'total_adv' :
    length(y_adv), 'correct_adv_counts' : adv_counts})
13: results_df.to_csv('/content/drive/MyDrive/ColabNotebooks/adv_results_labels.csv')

```



(a) confusion matrix 0.01



(b) confusion matrix 0.02

## Adversarial Robustness-toolbox - Counter Examples

Confusion Matrix for eps=0.03

True Labels \ Predicted Labels	0	1	2	3	4	5	6	7	8	9
0	0	55	0	0	0	918	0	0	0	0
1	0	21	0	0	0	1109	0	0	0	0
2	0	117	0	0	0	898	0	0	0	0
3	0	37	0	0	0	956	0	0	0	0
4	0	38	0	0	0	930	0	0	0	0
5	0	10	0	0	0	872	0	0	0	0
6	0	18	0	0	0	924	0	0	0	0
7	0	36	0	0	0	971	0	0	0	0
8	0	19	0	0	0	946	0	0	0	0
9	0	4	0	0	0	986	0	0	0	0

(a) confusion matrix 0.03

Confusion Matrix for eps=0.04

True Labels \ Predicted Labels	0	1	2	3	4	5	6	7	8	9
0	0	7	0	0	0	966	0	0	0	0
1	0	4	0	0	0	1126	0	0	0	0
2	0	17	0	0	0	998	0	0	0	0
3	0	5	0	0	0	988	0	0	0	0
4	0	9	0	0	0	959	0	0	0	0
5	0	1	0	0	0	881	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	6	0	0	0	1000	0	1	0	0
8	0	2	0	0	0	963	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0

(b) confusion matrix 0.04

Confusion Matrix for eps=0.05

True Labels \ Predicted Labels	0	1	2	3	4	5	6	7	8	9
0	0	1	0	0	0	972	0	0	0	0
1	0	1	0	0	0	1129	0	0	0	0
2	0	3	0	0	0	1012	0	0	0	0
3	0	1	0	0	0	992	0	0	0	0
4	0	5	0	0	0	963	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	1	0	0	0	1006	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0

(a) confusion matrix 0.05

Confusion Matrix for eps=0.1

True Labels \ Predicted Labels	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	1	0	0	0	967	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1006	0	1	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0

(b) confusion matrix 0.1

Confusion Matrix for eps=0.2

True Labels \ Predicted Labels	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	0	0	0	0	968	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1007	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0

(a) confusion matrix 0.2

Confusion Matrix for eps=0.3

True Labels \ Predicted Labels	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	0	0	0	0	968	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1007	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0

(b) confusion matrix 0.3

Confusion Matrix for eps=0.4

0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	0	0	0	0	968	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1007	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.4

Confusion Matrix for eps=0.5

0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	0	0	0	0	968	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1007	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(b) confusion matrix 0.5

0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	0	0	0	0	968	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1007	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.6

### 3. FSGM with corrected variables

---

#### Algorithm 3 Adversarial Example Generation and Analysis

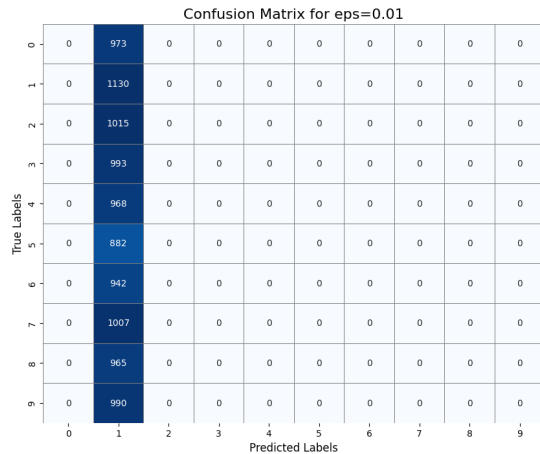
---

```

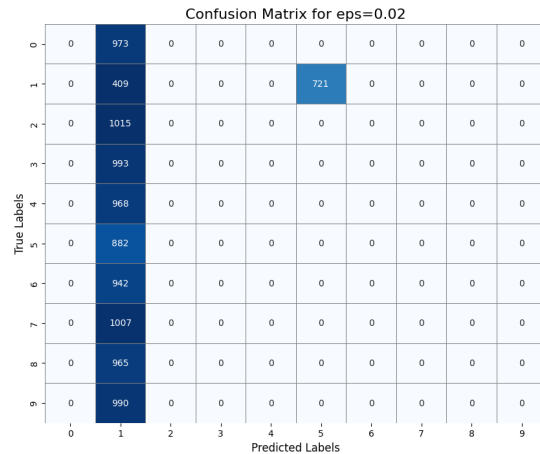
1: Input: classifier, correct_examples, correct_labels
2: Output: results_df, confusion matrices
3: eps_range  $\leftarrow$  [0.01, 0.02, ..., 0.6]
4: results_df  $\leftarrow$  DataFrame with columns ['eps', 'total_correct', 'total_adv', 'correct_adv_counts']
5: for eps  $\in$  eps_range do
6:   attack  $\leftarrow$  FastGradientMethod(classifier, eps)
7:   x_adv  $\leftarrow$  attack.generate(x=correct_examples, y=correct_labels)
8:   y_adv  $\leftarrow$  argmax(classifier.predict(x_adv), axis = 1)
9:   adv_counts  $\leftarrow$  count_unique(y_adv)
10:  cm  $\leftarrow$  confusion_matrix(correct_labels, y_adv)
11:  Save heatmap of cm to file with filename based on eps
12:  results_df.append({'eps' : eps, 'total_correct' : length(correct_labels), 'total_adv' :
    length(y_adv), 'correct_adv_counts' : adv_counts})
13: results_df.to_csv('/content/drive/MyDrive/ColabNotebooks/adv_results_labels.csv')

```

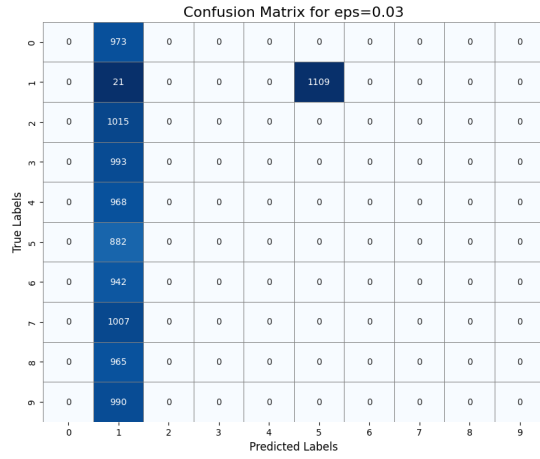
---



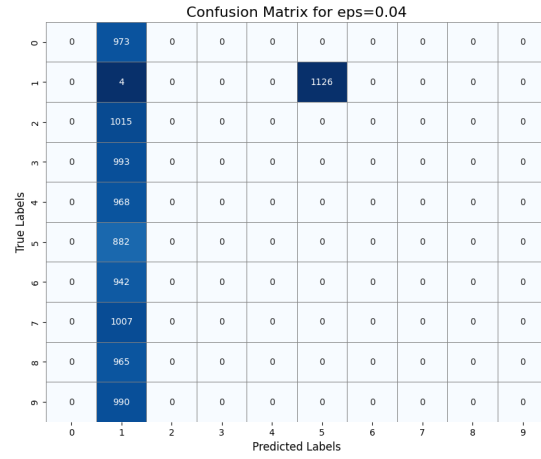
(a) confusion matrix 0.01 (output variable)



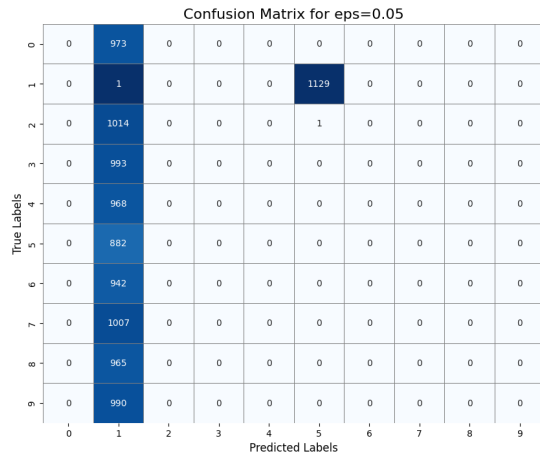
(b) confusion matrix 0.02 (output variable)



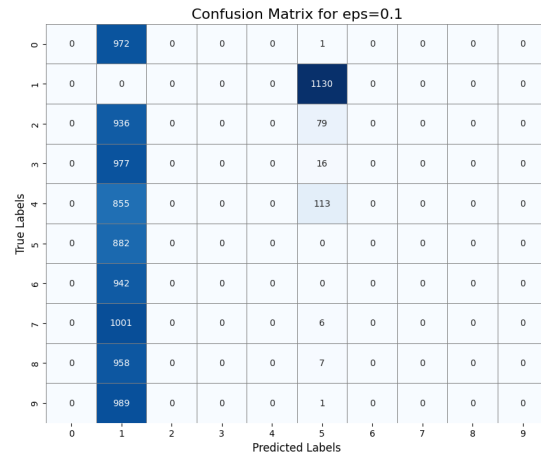
(a) confusion matrix 0.03 (output variable)



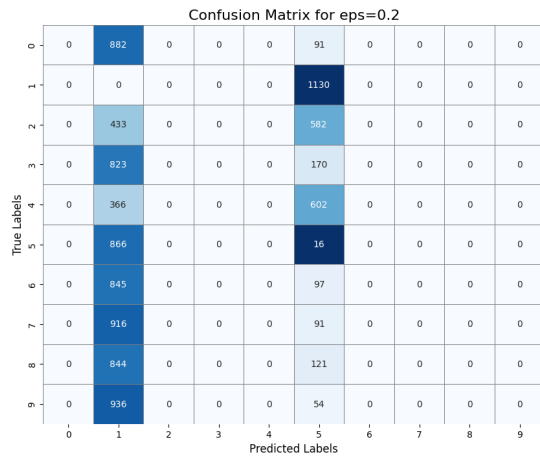
(b) confusion matrix 0.04 (output variable)



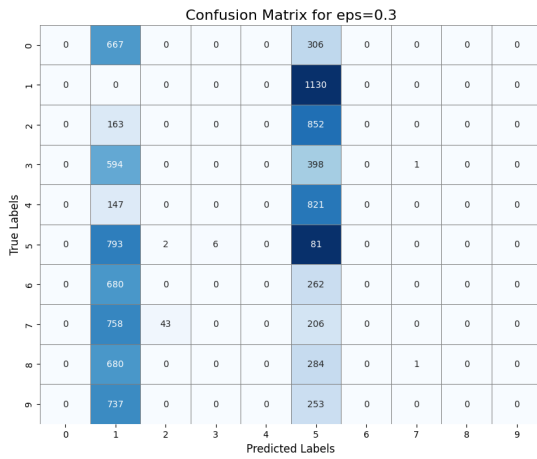
(a) confusion matrix 0.05 (output variable)



(b) confusion matrix 0.1 (output variable)



(a) confusion matrix 0.2(output variable)



(b) confusion matrix 0.3 (output variable)



Confusion Matrix for eps=0.4

True Labels \ Predicted Labels	0	1	2	3	4	5	6	7	8	9
0	0	470	0	0	0	503	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	50	0	0	0	965	0	0	0	0
3	0	398	2	0	0	593	0	0	0	0
4	0	56	0	0	0	912	0	0	0	0
5	0	707	10	10	0	155	0	0	0	0
6	0	497	0	0	0	445	0	0	0	0
7	0	512	176	0	0	319	0	0	0	0
8	0	497	2	0	0	462	0	4	0	0
9	0	514	2	0	0	474	0	0	0	0

Confusion Matrix for eps=0.5

True Labels \ Predicted Labels	0	1	2	3	4	5	6	7	8	9
0	0	297	0	0	0	676	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	15	0	0	0	1000	0	0	0	0
3	0	252	5	0	0	736	0	0	0	0
4	0	23	0	0	0	945	0	0	0	0
5	0	593	32	20	0	237	0	0	0	0
6	0	340	0	0	0	602	0	0	0	0
7	0	313	298	0	0	396	0	0	0	0
8	0	350	2	0	0	609	0	4	0	0
9	0	339	5	0	0	646	0	0	0	0

(a) confusion matrix 0.5 (output variable)

Confusion Matrix for eps=0.6

True Labels \ Predicted Labels	0	1	2	3	4	5	6	7	8	9
0	0	176	4	0	0	793	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	4	0	0	0	1011	0	0	0	0
3	0	155	12	0	0	826	0	0	0	0
4	0	11	0	0	0	957	0	0	0	0
5	0	455	57	45	0	325	0	0	0	0
6	0	210	0	0	0	732	0	0	0	0
7	0	180	383	0	0	444	0	0	0	0
8	0	263	5	0	0	694	0	3	0	0
9	0	221	11	0	0	758	0	0	0	0

(a) confusion matrix 0.6 (output variable)

# Adversarial Robustness-toolbox - Counter Examples

	eps	num	with.correct.Labels	without.correct.Labels
0	0.01	1	{1: 9865}	{1: 9865}
1	0.01	2	{1: 9865}	{1: 9865}
2	0.01	3	{1: 9865}	{1: 9865}
3	0.01	4	{1: 9865}	{1: 9865}
4	0.01	5	{1: 9865}	{1: 9865}
5	0.02	1	{1: 9144, 5: 721}	{1: 4068, 5: 5797}
6	0.02	2	{1: 9144, 5: 721}	{1: 4068, 5: 5797}
7	0.02	3	{1: 9144, 5: 721}	{1: 4068, 5: 5797}
8	0.02	4	{1: 9144, 5: 721}	{1: 4068, 5: 5797}
9	0.02	5	{1: 9144, 5: 721}	{1: 4068, 5: 5797}
10	0.03	1	{1: 8756, 5: 1109}	{1: 355, 5: 9510}
11	0.03	2	{1: 8756, 5: 1109}	{1: 355, 5: 9510}
12	0.03	3	{1: 8756, 5: 1109}	{1: 355, 5: 9510}
13	0.03	4	{1: 8756, 5: 1109}	{1: 355, 5: 9510}
14	0.03	5	{1: 8756, 5: 1109}	{1: 355, 5: 9510}
15	0.04	1	{1: 8739, 5: 1126}	{1: 51, 5: 9813, 7: 1}
16	0.04	2	{1: 8739, 5: 1126}	{1: 51, 5: 9813, 7: 1}
17	0.04	3	{1: 8739, 5: 1126}	{1: 51, 5: 9813, 7: 1}
18	0.04	4	{1: 8739, 5: 1126}	{1: 51, 5: 9813, 7: 1}
19	0.04	5	{1: 8739, 5: 1126}	{1: 51, 5: 9813, 7: 1}
20	0.05	1	{1: 8735, 5: 1130}	{1: 12, 5: 9853}
21	0.05	2	{1: 8735, 5: 1130}	{1: 12, 5: 9853}
22	0.05	3	{1: 8735, 5: 1130}	{1: 12, 5: 9853}
23	0.05	4	{1: 8735, 5: 1130}	{1: 12, 5: 9853}
24	0.05	5	{1: 8735, 5: 1130}	{1: 12, 5: 9853}
25	0.10	1	{1: 8512, 5: 1353}	{1: 1, 5: 9863, 7: 1}
26	0.10	2	{1: 8512, 5: 1353}	{1: 1, 5: 9863, 7: 1}
27	0.10	3	{1: 8512, 5: 1353}	{1: 1, 5: 9863, 7: 1}
28	0.10	4	{1: 8512, 5: 1353}	{1: 1, 5: 9863, 7: 1}
29	0.10	5	{1: 8512, 5: 1353}	{1: 1, 5: 9863, 7: 1}
30	0.20	1	{1: 6911, 5: 2954}	{5: 9865}
31	0.20	2	{1: 6911, 5: 2954}	{5: 9865}
32	0.20	3	{1: 6911, 5: 2954}	{5: 9865}
33	0.20	4	{1: 6911, 5: 2954}	{5: 9865}
34	0.20	5	{1: 6911, 5: 2954}	{5: 9865}
35	0.30	1	{1: 5219, 2: 45, 3: 6, 5: 4593, 7: 2}	{5: 9865}
36	0.30	2	{1: 5219, 2: 45, 3: 6, 5: 4593, 7: 2}	{5: 9865}
37	0.30	3	{1: 5219, 2: 45, 3: 6, 5: 4593, 7: 2}	{5: 9865}
38	0.30	4	{1: 5219, 2: 45, 3: 6, 5: 4593, 7: 2}	{5: 9865}
39	0.30	5	{1: 5219, 2: 45, 3: 6, 5: 4593, 7: 2}	{5: 9865}
40	0.40	1	{1: 3701, 2: 192, 3: 10, 5: 5958, 7: 4}	{5: 9865}
41	0.40	2	{1: 3701, 2: 192, 3: 10, 5: 5958, 7: 4}	{5: 9865}
42	0.40	3	{1: 3701, 2: 192, 3: 10, 5: 5958, 7: 4}	{5: 9865}
43	0.40	4	{1: 3701, 2: 192, 3: 10, 5: 5958, 7: 4}	{5: 9865}
44	0.40	5	{1: 3701, 2: 192, 3: 10, 5: 5958, 7: 4}	{5: 9865}
45	0.50	1	{1: 2522, 2: 342, 3: 20, 5: 6977, 7: 4}	{5: 9865}
46	0.50	2	{1: 2522, 2: 342, 3: 20, 5: 6977, 7: 4}	{5: 9865}
47	0.50	3	{1: 2522, 2: 342, 3: 20, 5: 6977, 7: 4}	{5: 9865}
48	0.50	4	{1: 2522, 2: 342, 3: 20, 5: 6977, 7: 4}	{5: 9865}
49	0.50	5	{1: 2522, 2: 342, 3: 20, 5: 6977, 7: 4}	{5: 9865}
50	0.60	1	{1: 1675, 2: 472, 3: 45, 5: 7670, 7: 3}	{5: 9865}
51	0.60	2	{1: 1675, 2: 472, 3: 45, 5: 7670, 7: 3}	{5: 9865}
52	0.60	3	{1: 1675, 2: 472, 3: 45, 5: 7670, 7: 3}	{5: 9865}
53	0.60	4	{1: 1675, 2: 472, 3: 45, 5: 7670, 7: 3}	{5: 9865}
54	0.60	5	{1: 1675, 2: 472, 3: 45, 5: 7670, 7: 3}	{5: 9865}

## 4. Multiple Attacks without corrected labels

---

### Algorithm 4 Adversarial Attack Evaluation

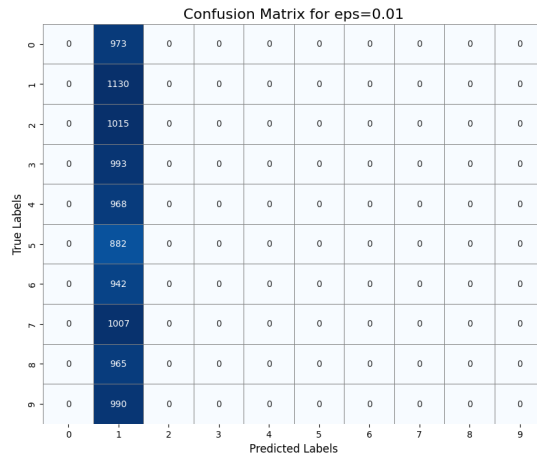
---

```

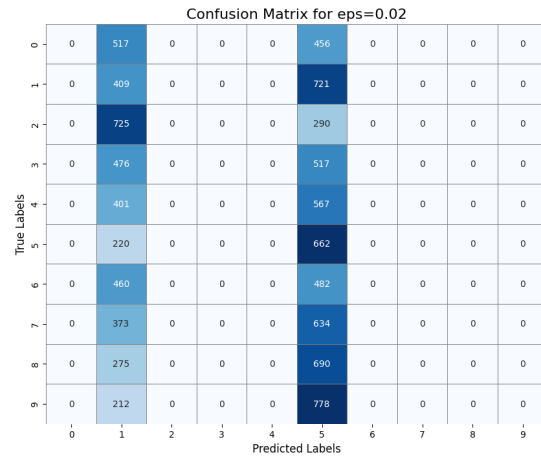
1: eps_range  $\leftarrow [0.01, 0.02, 0.03, \dots, 0.6]$ 
2: results_df  $\leftarrow$  DataFrame()
3: for eps  $\in$  eps_range do
4:   for attack_num  $\leftarrow$  1 to 5 do
5:     attack  $\leftarrow$  FastGradientMethod(classifier, eps)
6:     x_adv  $\leftarrow$  attack.generate(x=correct_examples)
7:     y_adv  $\leftarrow$  classifier.predict(x_adv)
8:     cm  $\leftarrow$  confusion_matrix(correct_labels, y_adv)
9:     Save heatmap of cm to file
10:    results_df.append({'eps': eps, 'attack_num': attack_num, ...})
11: results_df.to_csv('/content/drive/MyDrive/ColabNotebooks/adv_results.csv')
12: Print(results_df)

```

---



(a) confusion matrix 0.01 attack1



(b) confusion matrix 0.02 attack1

## Adversarial Robustness-toolbox - Counter Examples

Confusion Matrix for eps=0.03

0	0	55	0	0	0	918	0	0	0	0
1	0	21	0	0	0	1109	0	0	0	0
2	0	117	0	0	0	898	0	0	0	0
3	0	37	0	0	0	956	0	0	0	0
4	0	38	0	0	0	930	0	0	0	0
5	0	10	0	0	0	872	0	0	0	0
6	0	18	0	0	0	924	0	0	0	0
7	0	36	0	0	0	971	0	0	0	0
8	0	19	0	0	0	946	0	0	0	0
9	0	4	0	0	0	986	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.03 attack1

Confusion Matrix for eps=0.04

0	0	7	0	0	0	966	0	0	0	0
1	0	4	0	0	0	1126	0	0	0	0
2	0	17	0	0	0	998	0	0	0	0
3	0	5	0	0	0	988	0	0	0	0
4	0	9	0	0	0	959	0	0	0	0
5	0	1	0	0	0	881	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	6	0	0	0	1000	0	1	0	0
8	0	2	0	0	0	963	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(b) confusion matrix 0.04 attack1

Confusion Matrix for eps=0.05

0	0	1	0	0	0	972	0	0	0	0
1	0	1	0	0	0	1129	0	0	0	0
2	0	3	0	0	0	1012	0	0	0	0
3	0	1	0	0	0	992	0	0	0	0
4	0	5	0	0	0	963	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	1	0	0	0	1006	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.05 attack1

Confusion Matrix for eps=0.1

0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	1	0	0	0	967	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1006	0	1	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(b) confusion matrix 0.1 attack1

Confusion Matrix for eps=0.2

0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	0	0	0	0	968	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1007	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.2 attack1

Confusion Matrix for eps=0.3

0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	0	0	0	0	968	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1007	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(b) confusion matrix 0.3 attack1

Confusion Matrix for eps=0.4

0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	0	0	0	0	968	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1007	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.4 attack1

Confusion Matrix for eps=0.5

0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	0	0	0	0	968	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1007	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(b) confusion matrix 0.5 attack1

0	0	0	0	0	0	973	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	0	0	0	0	1015	0	0	0	0
3	0	0	0	0	0	993	0	0	0	0
4	0	0	0	0	0	968	0	0	0	0
5	0	0	0	0	0	882	0	0	0	0
6	0	0	0	0	0	942	0	0	0	0
7	0	0	0	0	0	1007	0	0	0	0
8	0	0	0	0	0	965	0	0	0	0
9	0	0	0	0	0	990	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

(a) confusion matrix 0.6 attack1

## 5. Multiple Attacks with corrected labels

---

### Algorithm 5 Adversarial Attack Evaluation

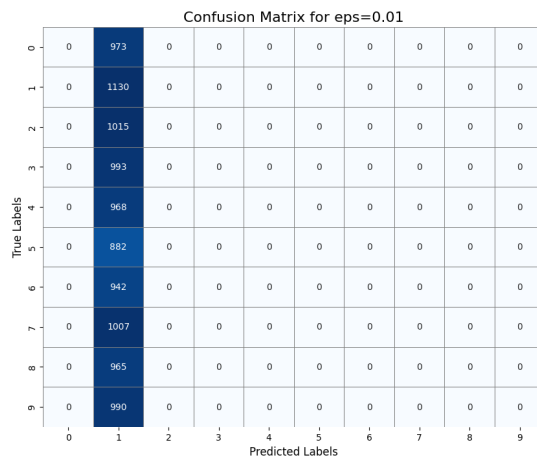
---

```

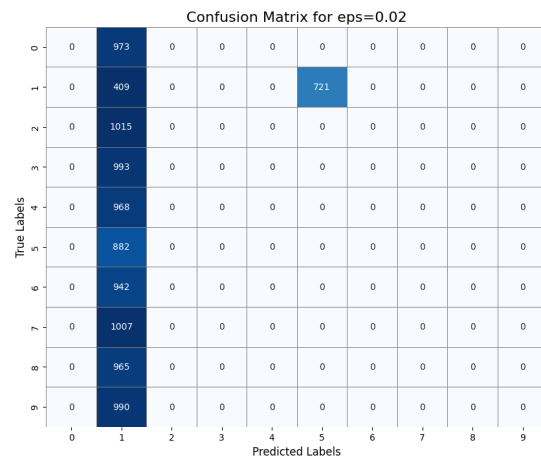
1: eps_range  $\leftarrow [0.01, 0.02, 0.03, \dots, 0.6]$ 
2: results_df  $\leftarrow$  DataFrame()
3: for eps  $\in$  eps_range do
4:   for attack_num  $\leftarrow$  1 to 5 do
5:     attack  $\leftarrow$  FastGradientMethod(classifier, eps)
6:     x_adv  $\leftarrow$  attack.generate(x=correct_examples, y=correct_labels)
7:     y_adv  $\leftarrow$  classifier.predict(x_adv)
8:     cm  $\leftarrow$  confusion_matrix(correct_labels, y_adv)
9:     Save heatmap of cm to file
10:    results_df.append({'eps': eps, 'attack_num': attack_num, ...})
11: results_df.to_csv('/content/drive/MyDrive/ColabNotebooks/adv_results.csv')
12: Print(results_df)

```

---



(a) confusion matrix 0.01 attack1 (output variable)



(b) confusion matrix 0.02 attack1 (output variable)

Confusion Matrix for eps=0.03

0	0	973	0	0	0	0	0	0	0	0
1	0	21	0	0	0	1109	0	0	0	0
2	0	1015	0	0	0	0	0	0	0	0
3	0	993	0	0	0	0	0	0	0	0
4	0	968	0	0	0	0	0	0	0	0
5	0	882	0	0	0	0	0	0	0	0
6	0	942	0	0	0	0	0	0	0	0
7	0	1007	0	0	0	0	0	0	0	0
8	0	965	0	0	0	0	0	0	0	0
9	0	990	0	0	0	0	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.03 attack1 (output variable)

Confusion Matrix for eps=0.04

0	0	973	0	0	0	0	0	0	0	0
1	0	4	0	0	0	1126	0	0	0	0
2	0	1015	0	0	0	0	0	0	0	0
3	0	993	0	0	0	0	0	0	0	0
4	0	968	0	0	0	0	0	0	0	0
5	0	882	0	0	0	0	0	0	0	0
6	0	942	0	0	0	0	0	0	0	0
7	0	1007	0	0	0	0	0	0	0	0
8	0	965	0	0	0	0	0	0	0	0
9	0	990	0	0	0	0	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(b) confusion matrix 0.04 attack1 (output variable)

Confusion Matrix for eps=0.05

0	0	973	0	0	0	0	0	0	0	0
1	0	1	0	0	0	1129	0	0	0	0
2	0	1014	0	0	0	1	0	0	0	0
3	0	993	0	0	0	0	0	0	0	0
4	0	968	0	0	0	0	0	0	0	0
5	0	882	0	0	0	0	0	0	0	0
6	0	942	0	0	0	0	0	0	0	0
7	0	1007	0	0	0	0	0	0	0	0
8	0	965	0	0	0	0	0	0	0	0
9	0	990	0	0	0	0	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.05 attack1 (output variable)

Confusion Matrix for eps=0.1

0	0	972	0	0	0	1	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	936	0	0	0	79	0	0	0	0
3	0	977	0	0	0	16	0	0	0	0
4	0	855	0	0	0	113	0	0	0	0
5	0	882	0	0	0	0	0	0	0	0
6	0	942	0	0	0	0	0	0	0	0
7	0	1001	0	0	0	6	0	0	0	0
8	0	958	0	0	0	7	0	0	0	0
9	0	989	0	0	0	1	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(b) confusion matrix 0.1 attack1 (output variable)

Confusion Matrix for eps=0.2

0	0	882	0	0	0	91	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	433	0	0	0	582	0	0	0	0
3	0	823	0	0	0	170	0	0	0	0
4	0	366	0	0	0	602	0	0	0	0
5	0	866	0	0	0	16	0	0	0	0
6	0	845	0	0	0	97	0	0	0	0
7	0	916	0	0	0	91	0	0	0	0
8	0	844	0	0	0	121	0	0	0	0
9	0	936	0	0	0	54	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.2 attack1 (output variable)

Confusion Matrix for eps=0.3

0	0	667	0	0	0	306	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	163	0	0	0	852	0	0	0	0
3	0	594	0	0	0	398	0	1	0	0
4	0	147	0	0	0	821	0	0	0	0
5	0	793	2	6	0	81	0	0	0	0
6	0	680	0	0	0	262	0	0	0	0
7	0	758	43	0	0	206	0	0	0	0
8	0	680	0	0	0	284	0	1	0	0
9	0	737	0	0	0	253	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(b) confusion matrix 0.3 attack1 (output variable)

Confusion Matrix for eps=0.4

0	0	470	0	0	0	503	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	50	0	0	0	965	0	0	0	0
3	0	398	2	0	0	593	0	0	0	0
4	0	56	0	0	0	912	0	0	0	0
5	0	707	10	10	0	155	0	0	0	0
6	0	497	0	0	0	445	0	0	0	0
7	0	512	176	0	0	319	0	0	0	0
8	0	497	2	0	0	462	0	4	0	0
9	0	514	2	0	0	474	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.4 attack1 (output variable)

Confusion Matrix for eps=0.5

0	0	297	0	0	0	676	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	15	0	0	0	1000	0	0	0	0
3	0	252	5	0	0	736	0	0	0	0
4	0	23	0	0	0	945	0	0	0	0
5	0	593	32	20	0	237	0	0	0	0
6	0	340	0	0	0	602	0	0	0	0
7	0	313	298	0	0	396	0	0	0	0
8	0	350	2	0	0	609	0	4	0	0
9	0	339	5	0	0	646	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(b) confusion matrix 0.5 attack1 (output variable)



Confusion Matrix for eps=0.6

0	0	176	4	0	0	793	0	0	0	0
1	0	0	0	0	0	1130	0	0	0	0
2	0	4	0	0	0	1011	0	0	0	0
3	0	155	12	0	0	826	0	0	0	0
4	0	11	0	0	0	957	0	0	0	0
5	0	455	57	45	0	325	0	0	0	0
6	0	210	0	0	0	732	0	0	0	0
7	0	180	383	0	0	444	0	0	0	0
8	0	263	5	0	0	694	0	3	0	0
9	0	221	11	0	0	758	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

True Labels

Predicted Labels

(a) confusion matrix 0.6 attack1 (output variable)