NUS CS-CS5562: Trustworthy Machine Learning October 12, 2023

Assignment 3

Lecturer: Reza Shokri *Student: Niharika Shrivastava A0254355A*

1 Warm-up

1.3 The Naive Membership Inference Attack

1. TP: 1484, TN: 1144, FP: 856, FN: 516
2. Attack accuracy of the naive membership inference attack: 65.7%

2 Membership Inference Attacks Using Loss Values

2.1 M_{loss} - Using Loss Values to Estimate a Single Threshold

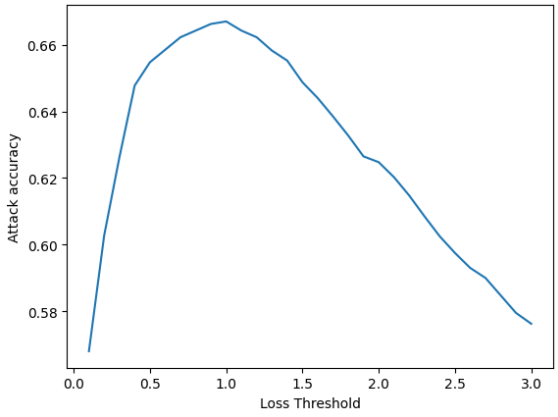
2.0.1 Initial Metrics

T_{loss}: 2.1, $Accuracy_{initial}$: 62.8%

2.0.2 Optimized Metrics

T_{loss}: 1.0, $Accuracy_{max}$: 66.7%

I set T_{loss} to range from $[0.1 - 3]$ by looking at the histogram for the train and test dataset loss values. Following this, I generate the membership labels and the corresponding attack accuracy for each T_{loss} and select the T_{loss} value which gives maximum attack accuracy.



2.2 $M_{loss,per\ class}$ - Using Loss Values to Estimate a Threshold for Each Class

The classes that show the clearest distinction in the loss values are Class 2 and 3 (Fig. 1).

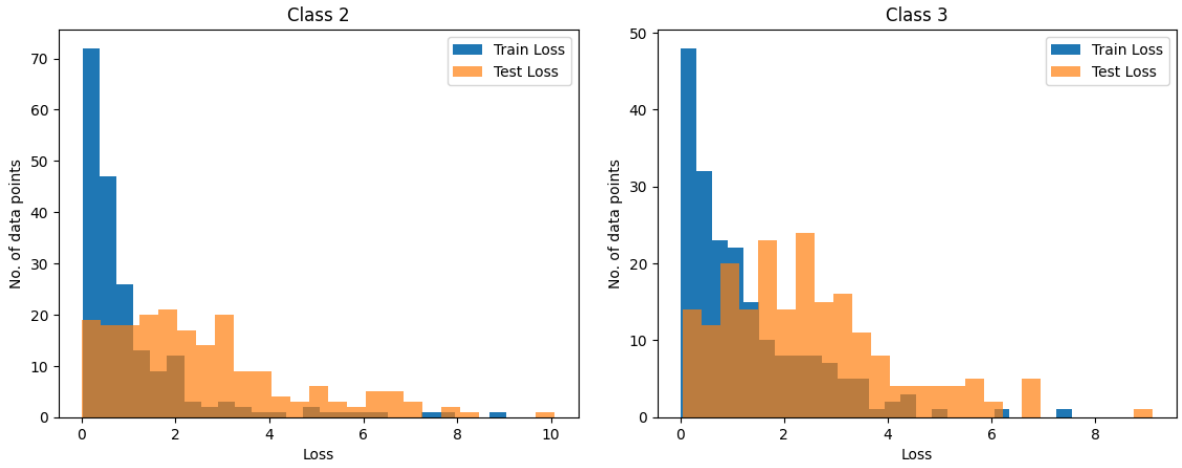


Figure 1: Distinction between losses

2.0.3 Initial Metrics

1. $T_{loss,classi} = \{1, 1, 1, 1, 1, 1, 1, 1, 1, 1\}$
2. Attack accuracy for each class = $\{60, 61.5, 64, 58.4, 60.5, 64.4, 64.75, 65.2, 63, 64\}$
3. Overall attack accuracy = 66.7%

2.0.4 Optimized Metrics

1. $T_{loss,classi} = \{0.4, 1.1, 1.0, 1.5, 0.6, 0.8, 0.4, 0.5, 0.8, 0.2\}$
2. % Attack accuracy for each class = $\{64, 70, 74.5, 69.5, 70.5, 67, 64.75, 67.75, 71, 66\}$
3. Overall attack accuracy = 68.5%

To maximize the overall accuracy, I find the best $T_{loss,classi}$ for each class using the method described in section 2.0.2, i.e., find the $T_{loss,classi}$ for each class that maximizes the accuracy of that class.

Compared with previous attacks, the attack accuracy is highest for $M_{loss,perclass}$. This is because we are considering a new piece of information that is the true class label of the input. We can see from the histograms that the distribution changes significantly depending on what class we are observing. If the distribution had been constant across all the classes then $M_{loss,perclass}$ accuracy would have been almost equal to M_{loss}.

2.3 Attacking a Model Trained With Regularization

Attack accuracy for M_{loss}: 57.35%
Attack accuracy for $M_{loss,perclass}$: 59.68%

Regularization generalizes the model performance on unseen data and not overfit on the training dataset. Therefore, the loss values on the train and test data are going to be very similar in case the model is regularized properly. In this case, choosing a T_{loss} naively will not work well, as the number of false positives will increase, thereby bringing down the accuracy of the attack significantly.

3 Attacking the MLaaS Setting

1. Randomly sample n images from $D_{other,challenge}$ for each class.
2. Split the above samples into n/2 train images and n/2 test images.
3. Train k shadow models using the above-mentioned split.
4. For each of the shadow model get the logits of their respective train and test dataset.
5. Append the true class labels to the logits array and assign label 1 (MEMBER) for train dataset and 0 (NON-MEMBER) for test dataset records.
6. Combine the above-mentioned datasets into $D_{train,attack}$. This will be used to train our attack model.
7. The Attack model's input is the logits along with the true class label. This is done because the prediction vector heavily depends on the true class label. Based on this, the attack model will classify it as 'MEMBER' or 'NON-MEMBER'.
8. Attack model is a neural network with a single dense layer of 64 units.
9. Once the attack model is trained, we pass the logits of $D_{eval,challenge}$ to generate the membership labels.

4 Testing Unintended Memorization

4.1 Assessing a CNN Classifier on CIFAR10

I set a threshold of 0.5 on memorization such that if the value is greater than 0.5, we can say with certainty that the model is memorizing. Otherwise, we cannot really comment on the model's memorizing power.

Based on this I selected 50 random data points and found out that for 17 data points the model is memorizing with an average memorization value of 0.761. For all of these

points, the predicted labels of h_θ and $h`_\theta$ do not match. Therefore, we can say that the model is memorizing to a large extent.

4.2 Assessing an LSTM Next Character Predictor on Shakespeare Dataset

Canary is: "The random sequence is XXXX". In the table below, we can see that there is a correlation between number of repetitions and the relative perplexity. As the number of repetitions k increase, the memorization also increases.

k	2	5	10	20	50
Perplexity	0.04261	1.583	3.3757	6.379	9.692

Table 1: Canary 1: ABCD

k	2	5	10	20	50
Perplexity	3.08	3.17	4.65	5.44	22.75

Table 2: Canary 2: AHJB