AI System Evaluation

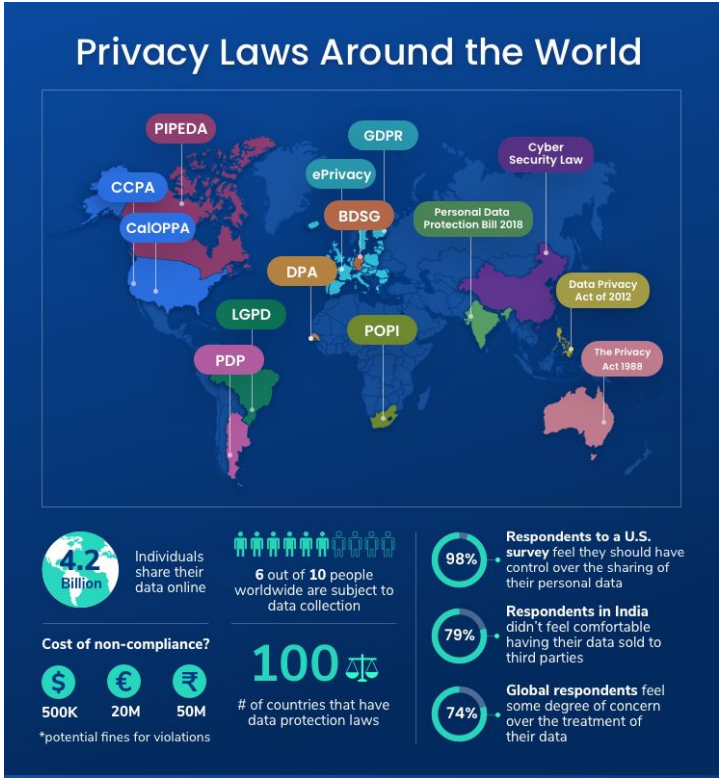
Week 9: Improving AI Privacy

Aug 23 - Week 1: 7-10	Introduction	
Aug 30 - Week 2: 7-10	AI Robustness	Exercise 1
Sep 06 - Week 3: 7-10	Improving AI Robustness	Exercise 2
Sep 13 - Week 4: 7-10	AI Backdoors	Exercise 3
Sep 20 - Week 5: 7-10	Mitigating AI Backdoors	Exercise 4; Project Proposal
Sep 27 - Week 6: 7-10	AI Fairness	Exercise 5
Oct 11 - Week 7: 7-10	Improving AI Fairness	Exercise 6
Oct 18 - Week 8: 7-10	AI Privacy	Exercise 7
Oct 25 - Week 9: 7-10	Improving AI Privacy	Exercise 8
Nov 01 - Week 10: 7-10	AI Interpretability	Project Due
Nov 08 - Week 11: 1-3	End-of-Term Exam	

Privacy

Privacy is ever more a relevant issue.

Machine learning relies on big data, which can be leaked directly or indirectly and cause privacy issues.



Outline

Avoiding data breaches

Avoiding indirect information exposure

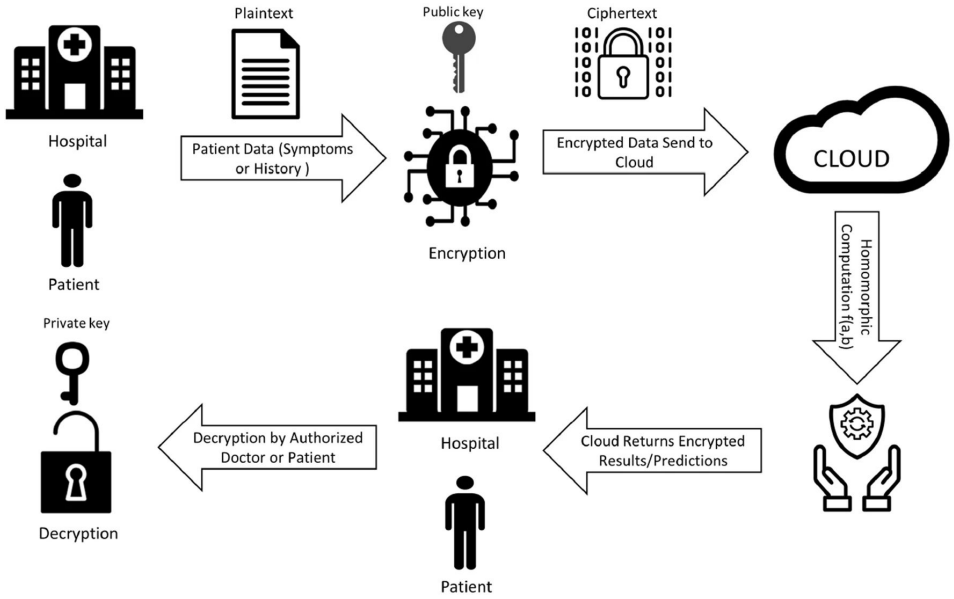
- Mitigating membership inference attacks
- Mitigating property inference attacks
- Mitigating model extraction attacks
- Mitigating model inversion attacks
- Mitigating model memorization attacks

Avoiding Dataset breaches

Apply your usual data protection practices. In addition, there are some techniques that are dedicated to machine learning.

Example 1:

Homomorphic encryption is a form of encryption with an additional evaluation capability for computing over encrypted data without access to the secret key. The result of such a computation remains encrypted.

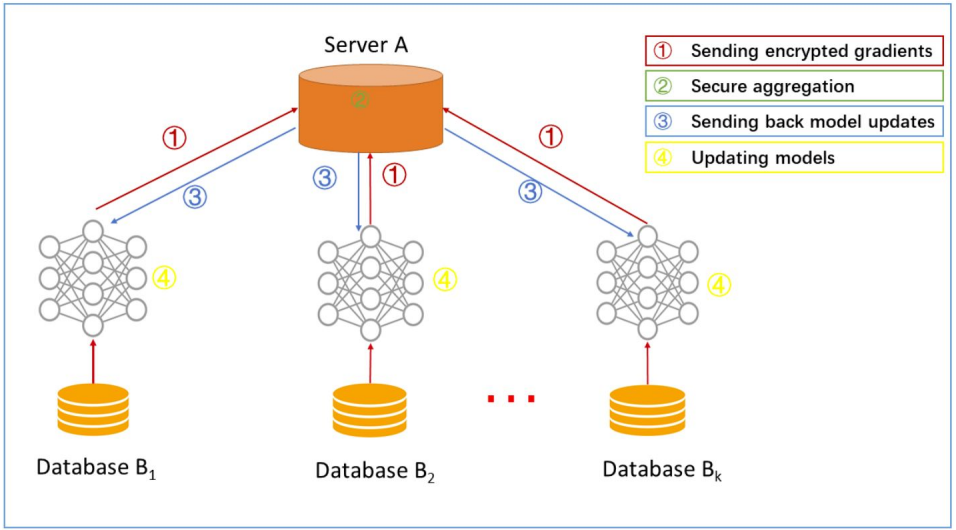


Homomorphic encryption is very slow.

Avoiding Dataset breaches: Examples

Example 2:

Federated learning (also known as collaborative learning) is a machine learning technique that trains an algorithm across multiple decentralized edge devices or servers holding local data samples, without exchanging them.



Discussion

If we apply homomorphic encryption and/or federated learning, are we immune from those indirect information exposures such as MIA?

Mitigating MIA

Confidence Score Masking

High-level idea

Many of the MIA relies on information provided by the confidence score (e.g., classifier-based MIA and Prediction Loss Based MIA). Maybe we simply hide the confidence score in some ways?

This way, the accuracy of the model is unaffected.

Approaches

The following masking schemes have proposed and experimented.

- Provide the label only
- Provide the top-K confidence only
- Round the confidence to a limited precision
- Add noise to the confidence before providing the confidence

Confidence Score Masking

Experimental Setup*

Attack: Classifier-based MIA

Defense: confidence score masking by providing labels only or limited confidence

Models: the purchase and Texas hospital-stay datasets (100 classes)

**Purchase dataset*

Purchase dataset	*Testing Accuracy*	*Attack Total Accuracy*	*Attack Precision*	*Attack Recall*
No Mitigation	0.66	0.92	0.87	1.00
Top $k = 3$	0.66	0.92	0.87	0.99
Top $k = 1$	0.66	0.89	0.83	1.00
Top $k = 1$ label	0.66	0.66	0.60	0.99
Rounding $d = 3$	0.66	0.92	0.87	0.99
Rounding $d = 1$	0.66	0.89	0.83	1.00

Hospital dataset	*Testing Accuracy*	*Attack Total Accuracy*	*Attack Precision*	*Attack Recall*
No Mitigation	0.55	0.83	0.77	0.95
Top $k = 3$	0.55	0.83	0.77	0.95
Top $k = 1$	0.55	0.82	0.76	0.95
Top $k = 1$ label	0.55	0.73	0.67	0.93
Rounding $d = 3$	0.55	0.83	0.77	0.95
Rounding $d = 1$	0.55	0.81	0.75	0.96

Membership Inference Attacks Against Machine Learning Models, S&P 2017.

Confidence Score Masking

High-level idea*

Add adversarial noise to the confidence score so that classifier-based MIA is likely to fail.

**Memguard: Defending against black-box membership inference attacks via adversarial examples. CCS 2019.*

Approach

Train a classifier X for classifier-based MIA

Given a confidence score vector, find some noise (through adversarial perturbation) so that *X's accuracy is reduced to a random choice* while the model N's accuracy is maintained.

Add the noise to the confidence score vector and return it to the user.

Confidence Score Masking

Approach

The noise is identified through optimization with the following objectives.

- The accuracy of MIA classifier is reduced to a random guess.
- The label remains the same.
- The confidence score distortion is minimized.

Exercise 1

Question 1: Why do we want to minimize the confidence score distortion?

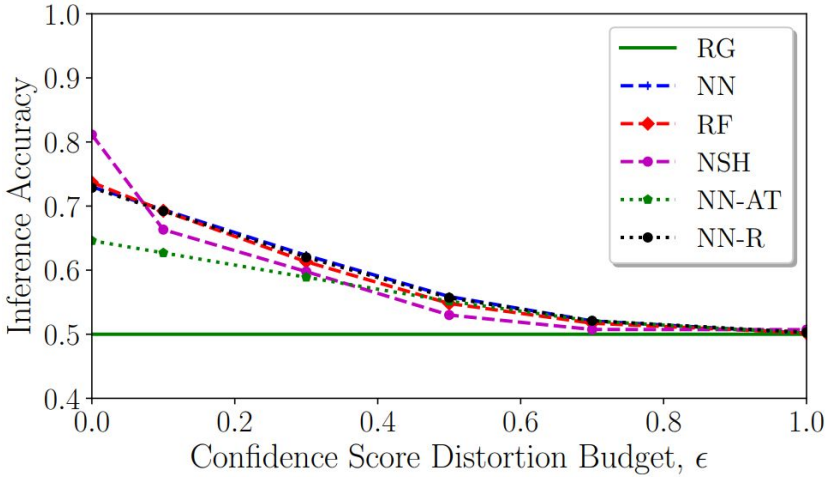
Question 2: Why don't we reduce the classifier's accuracy to nearly 0 (which is quite possible as you have seen in Week 2)?

Question 3: This method generates the noise based on a classifier which is likely different from the attacker's classifier. Why is it that it may still work?

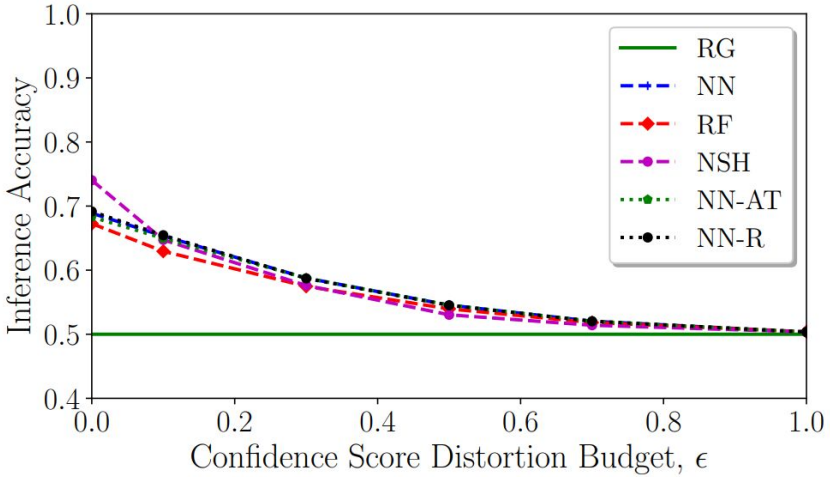
Confidence Score Masking: Performance

Against classifier-based MIA

NN-AT: neural network based MIA with adversarial training; NN-R: round confidence before NN-based MIA



(a) Location



(b) Texas100

Confidence Score Masking: Performance

Against metric-based MIA*

Classifier-based MIA

Four Metric-based MIA

dataset	Model Performance: using defense [20]?	training acc	test acc	Membership Inference Attacks: attack acc by [20]	attack acc (I_{corr})	attack acc (I_{conf})	attack acc (I_{entr})	attack acc (I_{Mentr})
Location30	no	100%	60.7%	**81.1%**	68.7%	76.3%	61.6%	78.1%
Location30	yes	100%	60.7%	50.1%	68.7%	**69.1%**	52.1%	68.8%
Texas100	no	99.95%	51.77%	74.0%	74.2%	79.0%	66.6%	**79.4%**
Texas100	yes	99.95%	51.77%	50.3%	**74.2%**	74.1%	54.6%	74.0%

**Systematic Evaluation of Privacy Risks of Machine Learning Models, USENIX 2021*

Exercise 2

Execute week9/exercise2/mentr.py to evaluate the performance of your mentr MIA attacker implemented last week (Week 5, exercise 5) if the top 3 confidences are provided by the model (i.e., the rest are masked off as 0.0).

1. Fix the bug
2. Modify the code so that only the top 2 and top 1 are provided by the model; and evaluate the performance of the mentr MIA attack.

Regularization

High-level idea

As we discussed, overfitting may be responsible for the effectiveness of MIA.

Techniques that aim to reduce overfitting thus could potentially help mitigate the risk of MIA.

How do we avoid overfitting then?

What is overfitting?

Overfitting occurs when a statistical model fits exactly against its training data.

Let D be the actual data distribution. D_r and D_e be the data distribution of the training and testing data. (Ideally, D_r and D_e and D are similar if not identical.)

Overfitting means that the model is trained to work optimally conditioned on D_r.

Regularization

High-level idea

We adopt different heuristics to make sure that the learned model is simple (i.e., the simpler a model is, the more likely it works well for D_e), based on the principle of Occam's razor.

Approaches: Regularization

L2-norm regularization
Dropout
Data augmentation
Model stacking
Early stopping
Label smoothing
...

A simpler model is likely to work for more data distributions.

Regularization

L2 Regularization

Training with the following objective:

minimize (loss + λ*(w_1^2 +w_2^2+w_3^2+...+w_n^2))

Dropout

Randomly drop out some percentage of neurons during training to simplify the model (and avoid overfitting).

We aim for models with fewer contributing neurons.

Data Augmentation

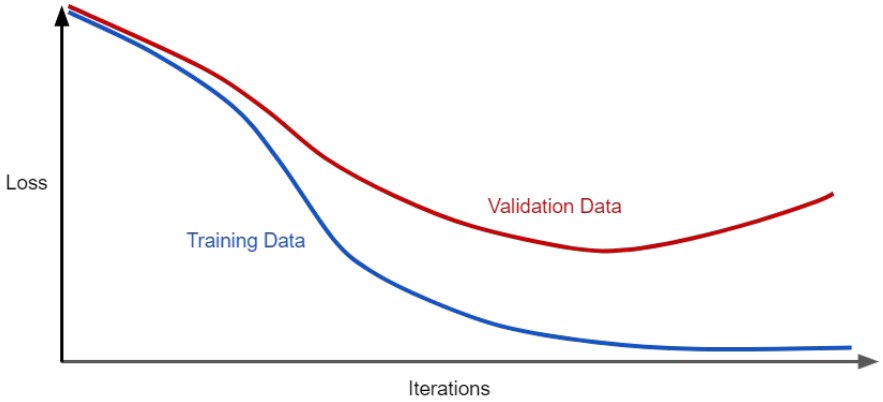
Add additional training samples obtained through various transformations, so that the model is forced **not** to learn from certain features.

Model Stacking

Use multiple models so that only features common in different models contribute.

We aim for models with fewer contributing features.

Regularization



Don’t be too sure.

Early Stopping

Stop training as soon as the generalization error (i.e., the difference between the accuracy on the training set and a holdout valuation set) increase.

Label Smoothing

Label Smoothing regularizes a model based on a softmax with k output values by replacing the hard 0 and 1 classification targets with targets of $\epsilon/(k-1)$ and $1-\epsilon$ respectively.

Regularization

The good news*

Traditional regularization indeed seems to reduce the risk of MIA.

The bad news

Regularization may also reduce the model accuracy.

**Membership Inference Attacks Against Machine Learning Models, S&P 2017.*

Purchase dataset	*Testing Accuracy*	*Attack Total Accuracy*	*Attack Precision*	*Attack Recall*
No Mitigation	0.66	0.92	0.87	1.00
Top $k = 3$	0.66	0.92	0.87	0.99
Top $k = 1$	0.66	0.89	0.83	1.00
Top $k = 1$ label	0.66	0.66	0.60	0.99
Rounding $d = 3$	0.66	0.92	0.87	0.99
Rounding $d = 1$	0.66	0.89	0.83	1.00
Temperature $t = 5$	0.66	0.88	0.86	0.93
Temperature $t = 20$	0.66	0.84	0.83	0.86
L2 $\lambda = 1e-4$	0.68	0.87	0.81	0.96
L2 $\lambda = 1e-3$	0.72	0.77	0.73	0.86
L2 $\lambda = 1e-2$	0.63	0.53	0.54	0.52

There seems to be a sweet spot?

Adversarial Regularization

High-level idea*

We have to two opposing objectives.

- Produce an accurate model, i.e., use as much information as possible;
- Produce a private model, i.e., use as little (sensitive) information as possible.

We train a model to satisfy both objectives.

**Machine Learning with Membership Privacy using Adversarial Regularization, CCS 2019*

Approach

Train the model by solving the following min-max optimization problem.

$$\min_N (L(N) + \lambda * \max_M G(M))$$

where N is the model to be trained; M is the classifier used in a classifier-based MIA; $\max_M G(M)$ is the maximum gain of the classifier-based MIA.

GAN?

Adversarial Regularization: Performance

Experiments

Dataset: Purchase100

Attack: classifier-based MIA

Observation: it provides privacy in the price of model accuracy.

λ	Training accuracy	Testing accuracy	Attack accuracy
0 (no defense)	100%	80.1%	67.6%
1	98.7%	78.3%	57.0%
2	96.7%	77.4%	55.0%
3	92.2%	76.5%	51.8%
10	76.3%	70.1%	50.6%

Is it worth the effort?

Adversarial Regularization: Performance

Experiments*

Dataset: Texas100

Attack: metric-based MIA

Observation: Its privacy protection is nearly as good as simple strategies such as early stopping.

attack methods	defense methods for Texas100 classifier		
	no defense	AdvReg [31]	early stopping
I_{conf} (class-independent)	64.7%	55.5%	55.8%
I_{conf} (class-dependent)	67.8%	58.6%	59.4%
I_{entr} (class-independent)	58.3%	52.9%	53.2%
I_{entr} (class-dependent)	60.2%	53.5%	54.0%
I_{Mentr} (class-independent)	64.8%	55.4%	55.9%
I_{Mentr} (class-dependent)	67.7%	58.6%	59.5%

**Systematic Evaluation of Privacy Risks of Machine Learning Models, USENIX 2021*

Knowledge Distillation

High-level idea*

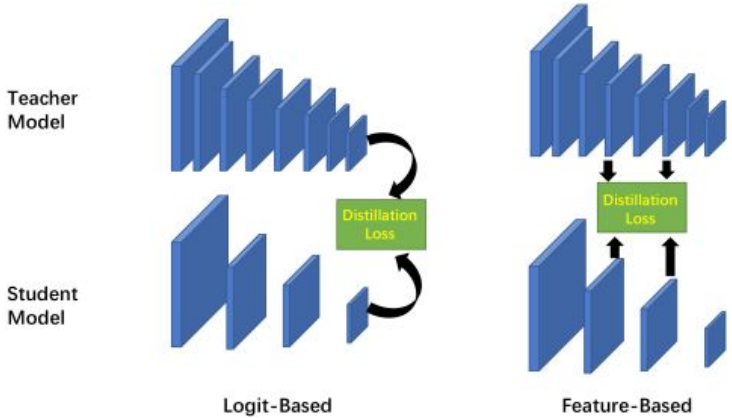
Through knowledge distillation, train a student model without using the original training data.

Naturally, the risk of leaking information on the training data is reduced.

**Membership Privacy for Machine Learning Models Through Knowledge Transfer, AAAI 2021.*

Knowledge Distillation

It is proposed as a way to reduce model size. It has the effect of pruning certain unnecessary details.



Knowledge Distillation

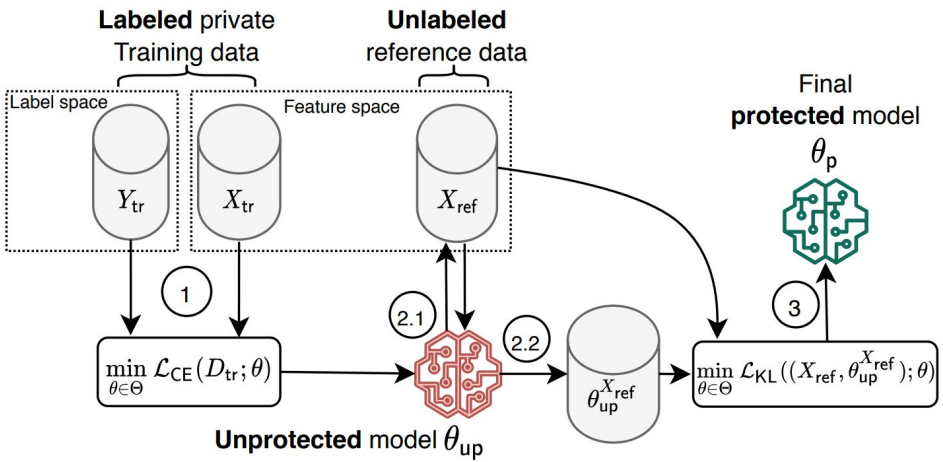
Approach

Step 1: Train an unprotected model N as per normal

Step 2: Generate a new training dataset

- Step 2.1: Identify suitable data from a set of unlabeled data
- Step 2.2: Label the identified data using N

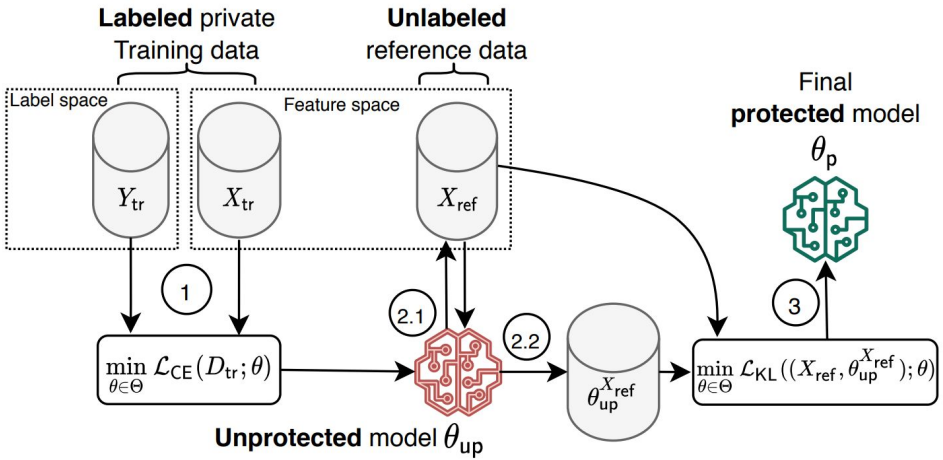
Step 3: Train a protected model M using the new training dataset



Knowledge Distillation

Step 2.1: Identify suitable data from a set of unlabeled data

The identified data should (1) be far away from the private training date; (2) have low entropy.



Can you intuitively explain why these are good ideas?

Knowledge Distillation: Performance

Experiments

Dataset: Purchase100, Texas100, CIFAR100 and CIFAR10

Attacks: classifier-based MIA, and metric-based MIA

Dataset and model	No defense					
	E_{gen}	A_{test}	A_{wb}	A_{bb}	A_{bl}	A_{nn}
P-FC	24.0	76.0	77.1	76.8	63.1	60.5
T-FC	51.3	48.7	84.0	82.2	76.1	71.9
C100-A	63.2	36.8	90.3	91.3	81.8	N/A
C100-D12	33.8	65.2	72.2	71.8	67.5	N/A
C100-D19	34.4	65.5	82.3	81.6	68.1	N/A
C10-A	32.5	67.5	77.9	77.5	66.4	N/A

Generalization error: Accuracy of testing - Accuracy of training

Knowledge Distillation: Performance

Knowledge distillation

Dataset and model	Adversarial regularization (AdvReg)						DMP						
	E_{gen}	A_{test}	Attack accuracy				E_{gen}	A_{test}	A^{+}_{test}	Attack accuracy			
			A_{wb}	A_{bb}	A_{bl}	A_{nn}				A_{wb}	A_{bb}	A_{bl}	A_{nn}
Purchase + FC	9.7	56.5	55.8	55.4	54.9	50.1	10.1	74.1	**+31.2%**	55.3	55.1	55.2	50.2
Texas + FC	6.1	33.5	58.2	57.9	54.1	50.8	7.1	48.6	**+45.1%**	55.3	55.4	53.6	50.0
CIFAR100 + Alexnet	6.9	19.7	54.3	54.0	53.5	N/A	6.5	35.7	**+81.2%**	55.7	55.6	53.3	N/A
CIFAR100 + DenseNet-12	5.5	26.5	51.4	51.3	52.8	N/A	3.6	63.1	**+138.1%**	53.7	53.0	51.8	N/A
CIFAR100 + DenseNet-19	7.2	33.9	54.2	53.4	53.6	N/A	7.3	65.3	**+92.6%**	54.7	54.4	53.7	N/A
CIFAR10 + Alexnet	4.2	53.4	51.9	51.2	52.1	N/A	3.1	65.0	**+21.7%**	51.3	50.6	51.6	N/A

Differential Privacy

High-level idea*

Add noise in a controlled manner such that we have some guarantee on the level of privacy that can be achieved.

Questions

How do we define or measure privacy?

How do we add noise in a privacy-preserving way?

How do we add noise into deep learning?

**Deep learning with differential privacy, CCS 2016.*

Differential Privacy

Defining Privacy

Let d be an arbitrary individual.

Let D be an arbitrary dataset.

Let $E = D \cup \{d\}$.

Let M be some way of obtaining information from a dataset.

We say M satisfies privacy if M(D) and M(E) are very similar.

Example

Trained on D, N_D predicts Jack has cancer with probability 0.55.

Trained on E, N_E predicts Jack has cancer with probability 0.57.

We probably can't tell whether "Jack has cancer" is in E.

What if N_E predicts Jack has cancer with probability 0.8?

Differential Privacy

Defining Privacy

We quantify the privacy loss as follows.

$$\log (\Pr(M(E)=r)/\Pr(M(D)=r))$$

We say that M satisfies ε-differential privacy if the privacy loss for any pair of D and E (that differ by one member) and any outcome r is bounded by some ε.

Example

From 0.55 to 0.57, the privacy loss is

$$\log(0.57/0.55) = 0.0357$$

From 0.55 to 0.8, the privacy loss is

$$\log(0.8/0.55) = 0.375$$

What if N_E predicts Jack has cancer with probability 0.2?

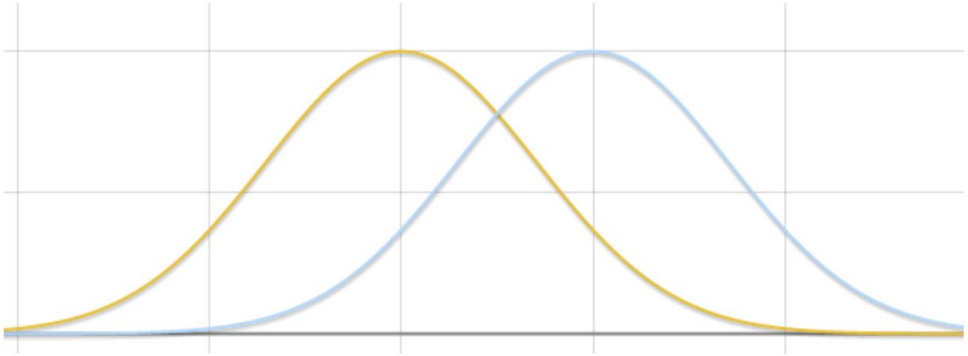
Differential Privacy for Deep Learning

Defining Privacy

We say that a deep learning model M satisfies (ε, δ)-differential privacy if

$Pr(M(D)=r) <= e^{\varepsilon}Pr(M(E)=r) + \delta$

where ε is the privacy budget and δ is a failure probability (which is there only to make the formal proof easier).



Intuitively, privacy means that the two distributions with or without certain data are close to each other.

Differential Privacy for Deep Learning

Question

How do we achieve (ε, δ)-differential privacy then?

Simple Answer

By systematically adding noises during the training process.

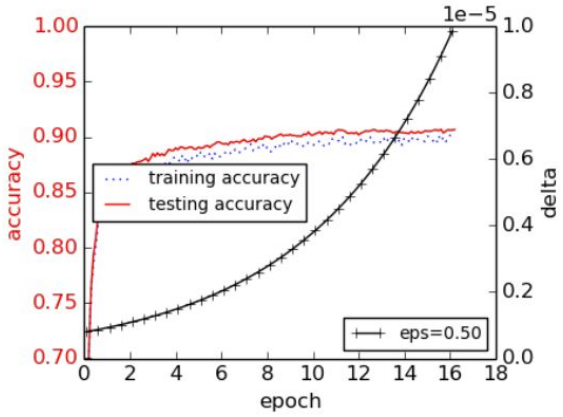
The algorithm is available in PyTorch.

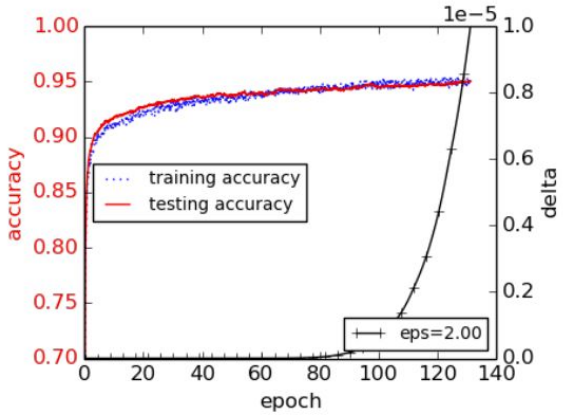
The real answer

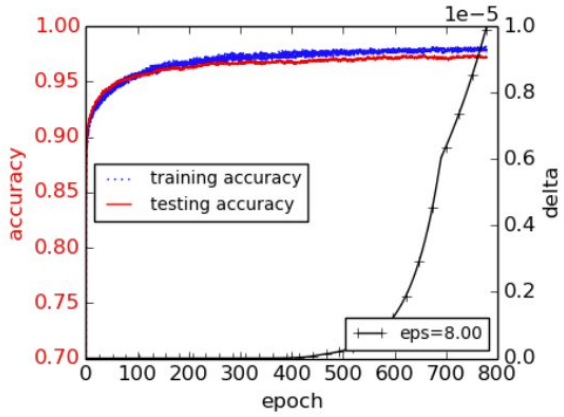
Initialize θ_0 randomly
for $t \in [T]$ **do**
Take a random sample L_t with sampling probability L/N
Compute gradient
For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$
Clip gradient
$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$
Add noise
$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$
Descent
$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$

Differential Privacy: Performance

MNIST







Smaller eps means more privacy and thus larger noise.

Differential Privacy: Performance

Systematic evaluation* (ε, δ) = (10, 10^{-5}); loss = 1 − withDP/withoutDP

Dataset	# Trainable Parameters	Classes	Train Loss w/ DP (%)	Test Loss w/ DP (%)	Non-Private Vulnerability (%)
MNIST	71,010	10	5.09	3.51	53.11
CIFAR-10	83,978	10	55.90	14.34	72.58
CIFAR-100	107,108	100	82.66	49.34	74.04
CIFAR-10 (with TL)	1,036,810	10	51.00	26.59	72.94
CIFAR-100 (with TL)	1,071,460	100	85.60	58.39	89.08

Is the loss acceptable?

**Effects of differential privacy and data skewness on membership inference vulnerability. TPS-ISA 2019.*

Exercise 3

week9/exercise3/dp.py is a program which applies differential privacy when training a MNIST model. Vary the value of ε (such as 10, 1, 0.1) and δ (0.1, 0.01, 0.001) and see how it affects the test accuracy and the attack success rate of the mentr-based MIA attack.

ε	δ	test accuracy	MIA accuracy
NA	NA		
10	0.1		
10	0.01		

Mitigating Property Inference Attacks

Mitigating Property Inference Attack

Causes of Property Inference Attacks

Property inference is possible even with well-generalized models.

We have little information about what makes them possible and under which circumstances they appear to be effective.

Any idea how it might be the case?

Results*

Differential privacy does not seem to offer protection against property inference attacks.

Regularization had an adverse effect and actually made the attacks stronger.

**Exploiting Unintended Feature Leakage in Collaborative Learning, S&P 2019.*

Mitigating Model Extraction Attacks

Mitigating Model Extraction Attacks

High-level idea

With a sufficient number of queries, model extraction attacks are just like model learning, i.e., there is no preventing it.

The goal is thus to make it harder to “learn”.

Approaches

Do not provide an API.

Provide the label only.

Introduce noise in the prediction confidence.

Refuse to answer weird queries.

Limit the number of queries from malicious users.

Detecting Model Extraction

High-level idea*

Detect those queries which are likely designed for model extraction and prevent them.

The detection is based on the assumption that model queries that try to explore decision boundaries will have a different distribution than the normal ones.

**PRADA: protecting against DNN model stealing attacks. EuroS&P, 2019.*

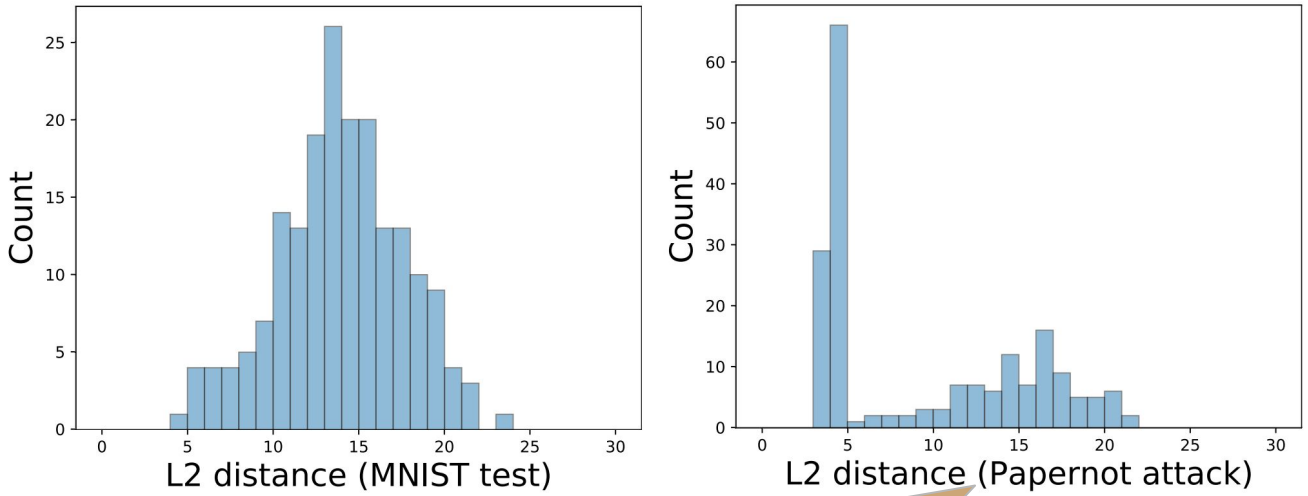
Attacking queries vs normal queries

Normal queries: The distance between two randomly selected points from a totally bounded space (e.g., a cube) often fits a normal (Gaussian) distribution. Normal queries thus form a normal (Gaussian) distribution.

Attacking queries: They are designed to extract maximal information and thus more likely not a normal distribution.

Detecting Model Extraction

Normal queries vs model extracting queries



The minimum L2 distance between a query and preceding queries.

Detecting Model Extraction

Question

How do we test the normality of the queries then?

Answer

The Shapiro-Wilk test

Approach

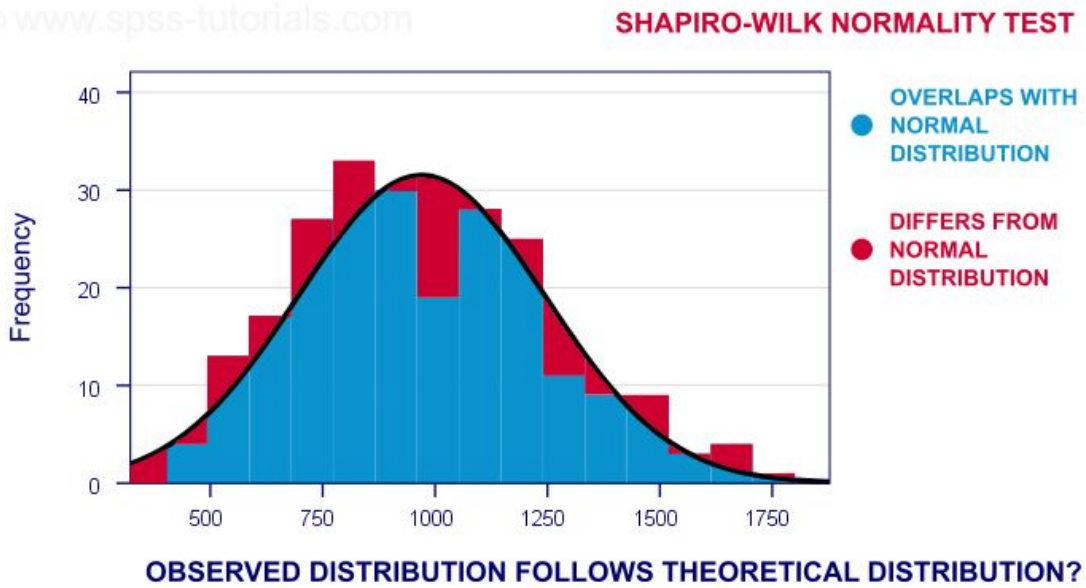
Wait until a client a sufficient number of queries (e.g., 100 times).

Optionally remove outliers.

Compute W according to the Shapiro-Wilk test.

Compare W with a threshold.

Shapiro-Wilk Test



week9/shapiro.py

Discussion

What would you do to prevent your model from being stolen?

Mitigating Model Extraction: Watermarking

High-level idea*

Train the model with some watermark (a.k.a. backdoor).

The extracted model may contain the backdoor.

We can then prove that the attacker stole the model.

Discussion

What do you think of this defense?

With your knowledge on backdoor attacks, will it work? And how would you counter this defense?

Does it achieve the goal?

**Thieves on Sesame Street! Model Extraction of BERT-based APIs, ICLR 2020.*

Mitigating Model Inversion Attacks

Mitigating Model Inversion Attacks

Causes of model inversion attacks

Overfitting may be a cause of model inversion attacks.

Lack of data variety may be a reason as well.

Out-of-distribution samples may be more vulnerable to model inversion.

Approaches

Methods that prevent overfitting (refer to Slide 16)

Differential privacy (refer to Slide 29)

Provide less informative prediction (i.e., confidence score masking, refer to Slide 9)

Mitigating Model Inversion Attacks

Experimental Setup

Insert canaries such as “My social security number is 078-05-1120.” into a training dataset.

Train models such as LSTM.

Evaluate the effect of methods on preventing memorization.

Results

Traditional regulation, e.g., drop out and quantization: the canaries can still be extracted by the attacker.

Differential Privacy: Even with a vanishingly-small amount of noise, and values of ε that offer no meaningful theoretical guarantees, training with differential privacy prevents the attacker from extracting the canaries effectively.

Mitigating Model Memorization Attack

Mitigating Model Memorization Attack

Recall: Black-box setting

Let D be the data to be memorized. Assume there are n classes.

For every log2n bits of D, generate a random input (e.g., images with one non-zero pixel value or random sentence) using a deterministic algorithm and label it with the i-th class (where i is the value of the log2n bits).

Train the model with the training data and the additional data.

Mitigation

Use an anomaly detection algorithm to detect abnormal samples (either during training or inference) and filter them.

What else you can think of?

Exercise 4

For each of the following white-box model memorization attacks, suggest a way of mitigating them.

- Least significant bit encoding: use the least significant bits of each parameter to memorize the data
- Correlated value encoding: add a loss to encourage “memorizing” data during training
- Sign encoding: use the sign of each parameter to memorize the data

Conclusion

Apply standard data protection practices to prevent direct data breaches.

Knowledge distillation is perhaps the most effective approach for mitigating MIA.

Apply differential privacy with some relatively large ε.

Differential privacy often brings high level of accuracy drop if you would prefer to have some theoretical guarantee.

Exercise 5

Conduct a model extraction attack on the model week9/exercise5/MNIST.pt. You can use the 10% of the original train set to form your queries. Compare the performance with the following three settings.

1. Only the label is provided (finish in-class if you can);
2. The full confidence is provided;
3. the confidence is rounded off to 2 decimal places;

Evaluate the performance of the attack using the fidelity (the rate of agreement between the extracted model and the original model) and test accuracy.

Aug 23 - Week 1: 7-10	Introduction	
Aug 30 - Week 2: 7-10	AI Robustness	Exercise 1
Sep 06 - Week 3: 7-10	Improving AI Robustness	Exercise 2
Sep 13 - Week 4: 7-10	AI Backdoors	Exercise 3
Sep 20 - Week 5: 7-10	Mitigating AI Backdoors	Exercise 4; Project Proposal
Sep 27 - Week 6: 7-10	AI Fairness	Exercise 5
Oct 11 - Week 7: 7-10	Improving AI Fairness	Exercise 6
Oct 18 - Week 8: 7-10	AI Privacy	Exercise 7
Oct 25 - Week 9: 7-10	Improving AI Privacy	Exercise 8
Nov 01 - Week 10: 7-10	AI Interpretability	Project Due
Nov 08 - Week 11: 1-3	End-of-Term Exam	