Title: Privacy-preserving machine learning with tensor networks

Speakers: Alejandro Pozas Kerstjens

Series: Quantum Foundations

Date: March 04, 2024 - 11:30 AM

URL: https://pirsa.org/24030105

Abstract: In this talk, I will argue and practically illustrate that insights in quantum information, concretely coming from the tensor network representations of quantum many-body states, can help in devising better privacy-preserving machine learning algorithms. In the first part, I will show that standard neural networks are vulnerable to a type of privacy leak that involves global properties of the data used for training, thus being a priori resistant to standard protection mechanisms. In the second, I will show that tensor networks, when used as machine learning architectures, are invulnerable to this vulnerability. The proof of the resilience is based on the existence of canonical forms for such architectures. Given the growing expertise in training tensor networks and the recent interest in tensor-based reformulations of popular machine learning architectures, these results imply that one may not have to be forced to make a choice between accuracy in prediction and ensuring the privacy of the information processed when using machine learning on sensitive data.

Zoom link

Privacy-preserving machine learning with tensor networks

Alex Pozas-Kerstjens, Senaida Hernández-Santana, José Ramón Pareja Monturiol, Marco Castrillón López, Giannicola Scarpa, Carlos E. González-Guillén, David Pérez-García

UNIVERSITÉ DE GENÈVE

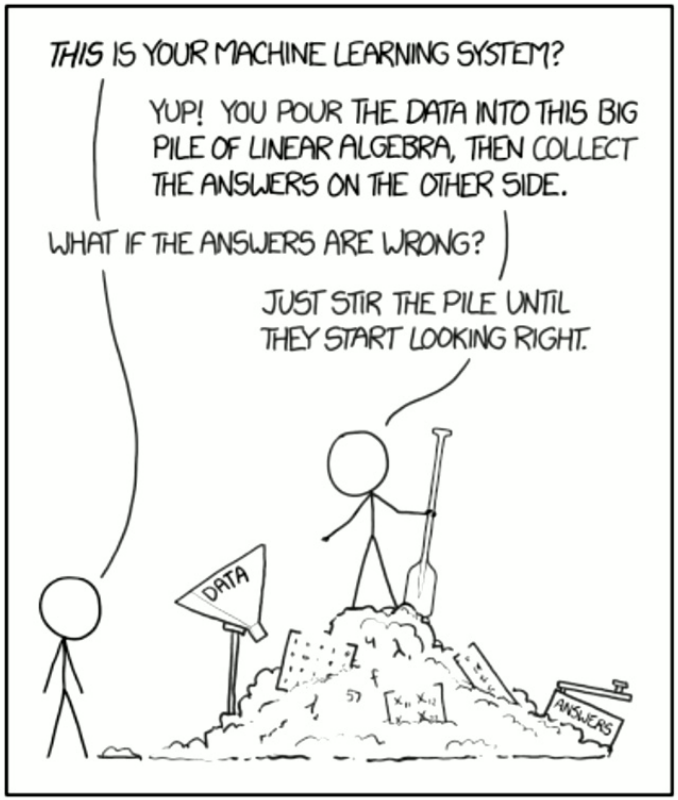
CONSTRUCTOR INSTITUTE

How it started

How it's going

Well, let's see

When I entered ML



https://xkcd.com/1838/

REUTERS World Business Markets Sustainability Legal Breakingviews Technology Investig

Litigation | Attorney Analysis | Data Privacy | Corporate Counsel

The privacy paradox with AI

By **Gai Sher** and **Ariela Benchlouch**

October 31, 2023 6:15 PM GMT+1 · Updated 4 months ago

Commentary | Attorney Analysis from Westlaw Today, a part of Thomson Reuters.

SEARCH FORTUNE Sub

Home News Tech Finance Leadership Well Recommends Fortune 500

TECH · THE FUTURE OF WORK

AI Has a Big Privacy Problem and Europe's New Data Protection Law Is About to Expose It

BY DAVID MEYER

CIVICUS Home Who We Are What We Do Get Involved Media

AI: 'The biggest challenges are the biases and lack of transparency of algorithms'

Interviews / 24 August 2023

INFORMATION WEEK NEWSLETTER SIGN-UP

Leadership Resilience ML & AI Data Sustainability Infrastructure Software More

OpenAI's ChatGPT Violates GDPR: Italy's Data Watchdog Says

The watchdog group that previously enforced a temporary ChatGPT ban says OpenAI must respond to allegations of data privacy breaches as Europe moves closer to passing sweeping AI rules.

Shane Snider January 30, 2024 2 Min Read

Editor's Choice

Office of the Privacy Commissioner of Canada | Commissariat à la protection de la vie privée du Canada | Search priv.gc.ca

For individuals | For businesses | For federal institutions | Report a concern | OPC actions and decisions

Home ➔ Blog

Privacy Tech-Know blog: When worlds collide – The possibilities and limits of algorithmic fairness (Part 1)

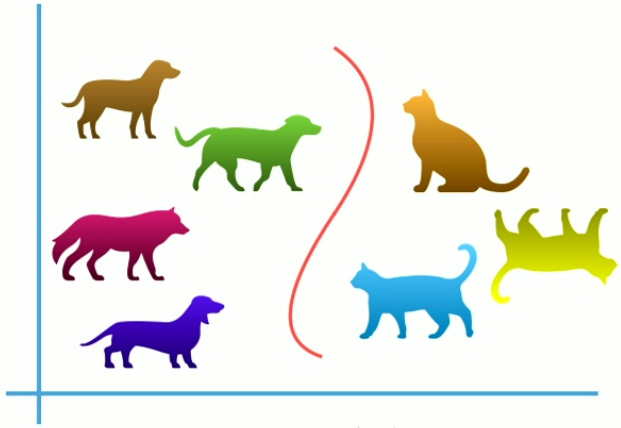
David Weinkauf, April 5, 2023 - Private sector, Guidance for organizations, Privacy Tech-Know Blog, Technology

About the blog

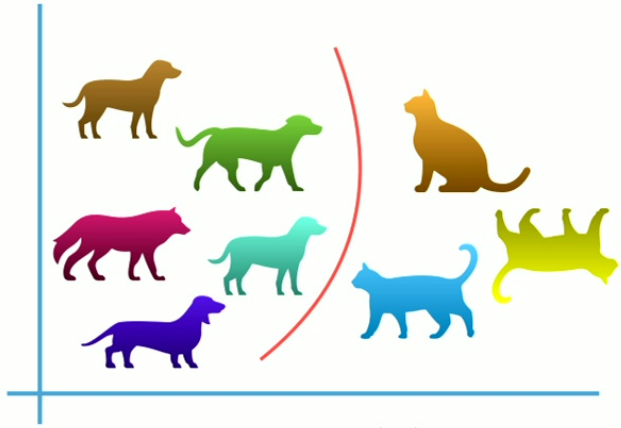
In this talk

How physics can help in ML privacy

Privacy in Supervised Learning



$y = f_\theta(x)$

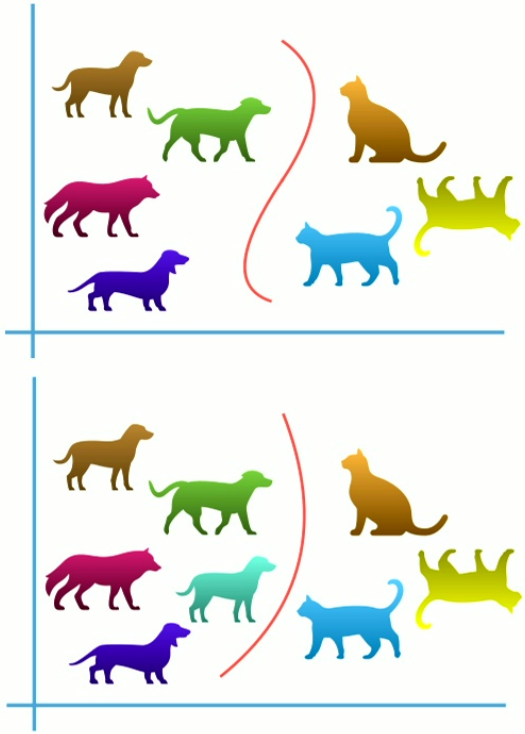


$y = f_{\theta'}(x)$

The decision boundary can depend on the data used for training. Can we extract this?

A privacy concern: Membership inference

Acces to a model (via its parameters or queries to it) can reveal the presence of a point in the training dataset.



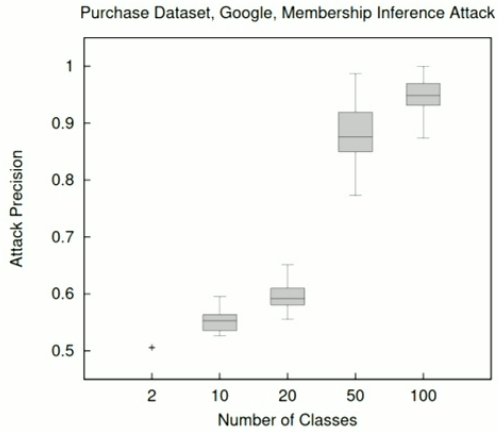
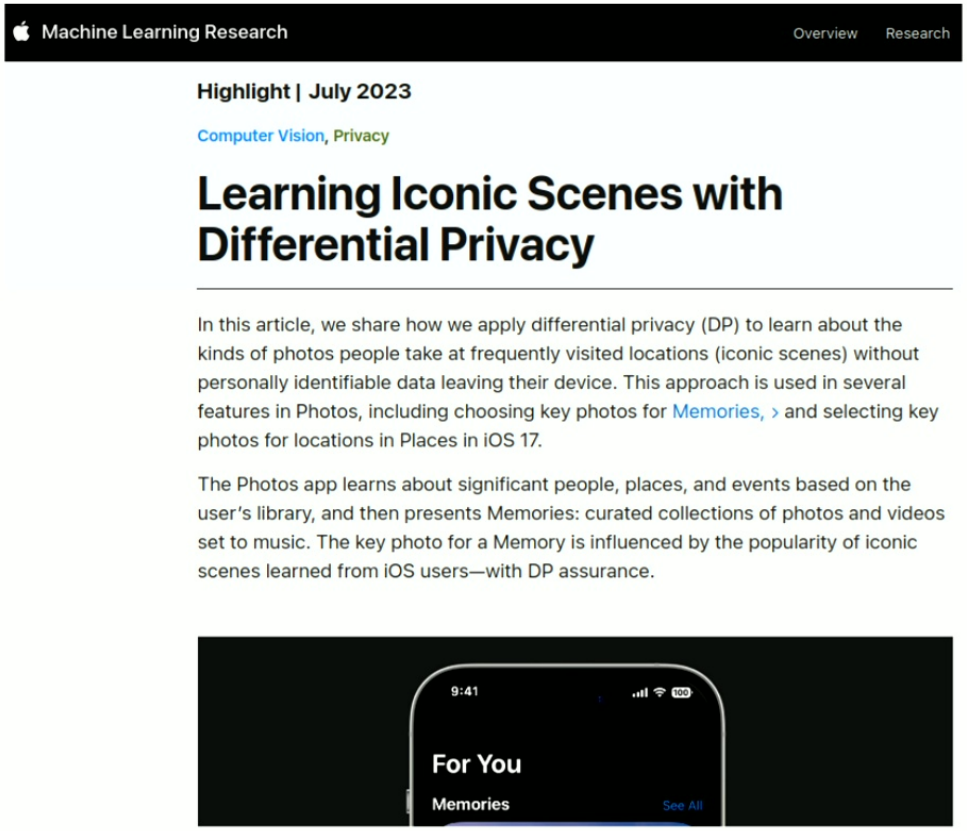


Fig. 10: Precision of the membership inference attack against different purchase classification models trained on the Google platform. The boxplots show the distribution of precision over different classification tasks (with a different number of classes).

Shokri, Stronati, Song, Shmatikov, DOI:10.1109/SP.2017.41

Protecting privacy in ML: differential privacy



https://machinelearning.apple.com/research/scenes-differential-privacy

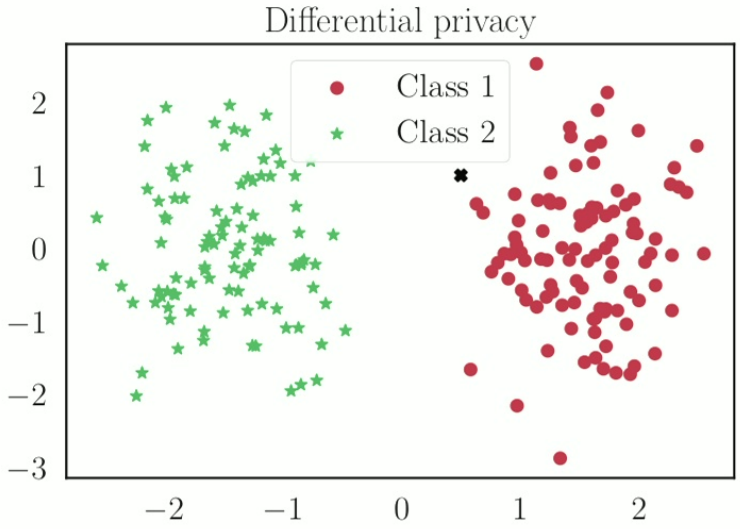
In summary: add noise

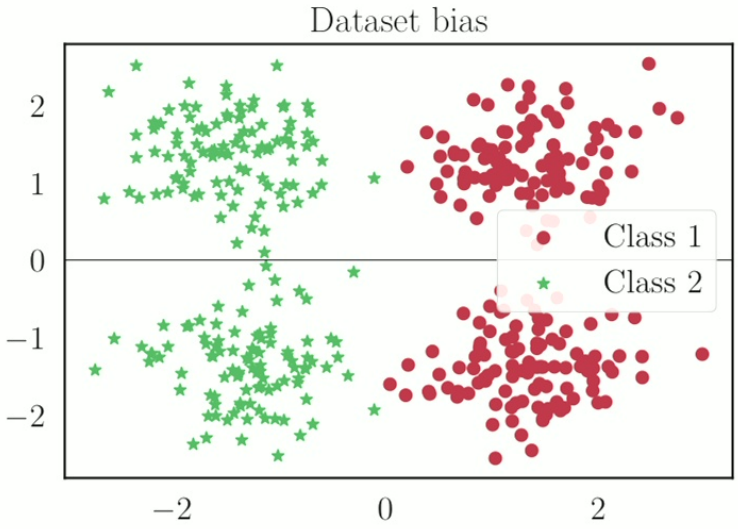
Privacy-utility tradeoff

- Slower training
- Cost in performance

Has disparate effect in minorities (arXiv:1905.12101)

A new privacy concern





Example: morbidities in medical records
Not clear that DP helps here

The vulnerability in a toy model

$\boldsymbol{x} = (x_{\text{rel}}, x_{\text{irr}})$

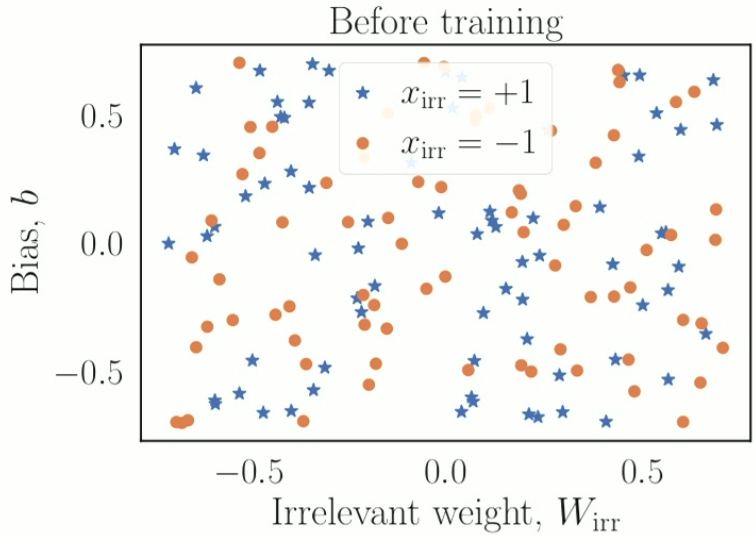
$\text{NN}(x) = \tanh\left(W_{\text{rel}} x_{\text{rel}} + W_{\text{irr}} x_{\text{irr}} + b\right)$

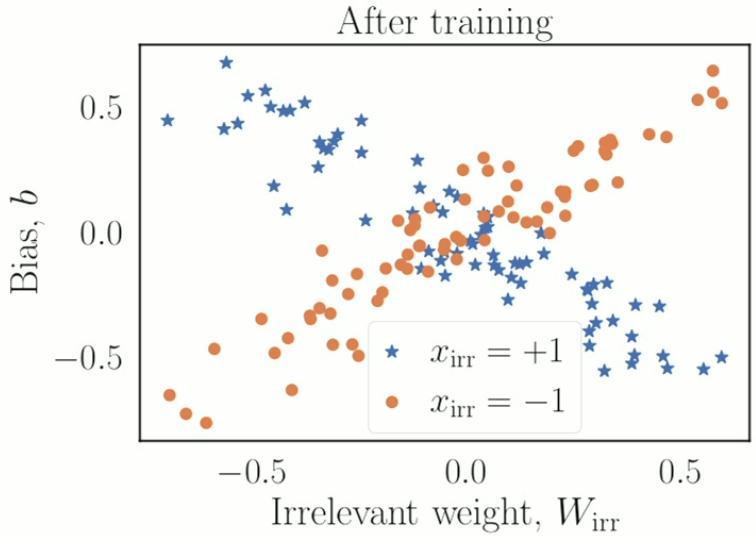
Goal: learn $y(\boldsymbol{x}) = \text{sign}(x_{\text{rel}})$

Test accuracy $> 95\%$

Attack: $\tilde{x}_{\text{irr}} = -\text{sign}(W_{\text{irr}} b)$

Attack accuracy $> 85\%$





For <u>any</u> loss function, $\partial_{W_{\text{irr}}} \mathcal{L} = x_{\text{irr}} \partial_b \mathcal{L}$.

The vulnerability in real-world data

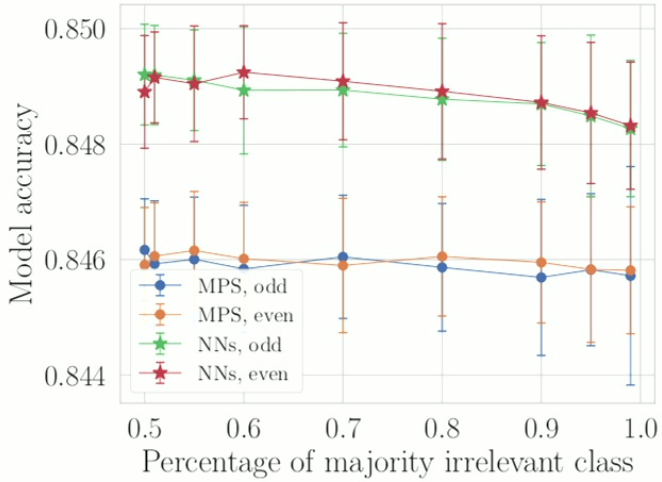
Model

Predict COVID-19 infection outcome ($d_{\text{out}} = 2$) given demographics & symptoms ($d_{\text{in}} = 9$).

Five-layer NN, 614 parameters.

x_{irr}: parity of the date of registration of the case.

Test accuracy: $\sim 84.9\%$, irrespective of x_{irr}.



https://global.health

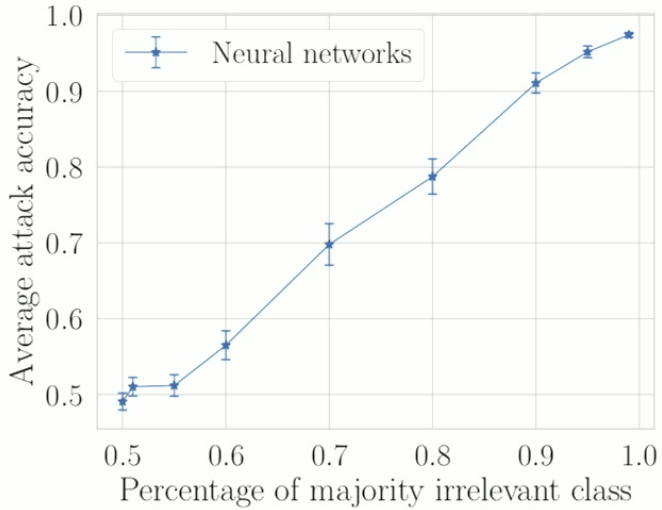
The vulnerability in real-world data

Model

Predict COVID-19 infection outcome ($d_{\text{out}} = 2$) given demographics & symptoms ($d_{\text{in}} = 9$).
Five-layer NN, 614 parameters.
x_{irr}: parity of the date of registration of the case.
Test accuracy: $\sim 84.9\%$, irrespective of x_{irr}.

Over-powerful adversary

Has access to labeled data of models and bias in the corresponding training set.
Trains a logistic regressor ($\#_{\text{params}} = 615$), evaluates on models not used in training.

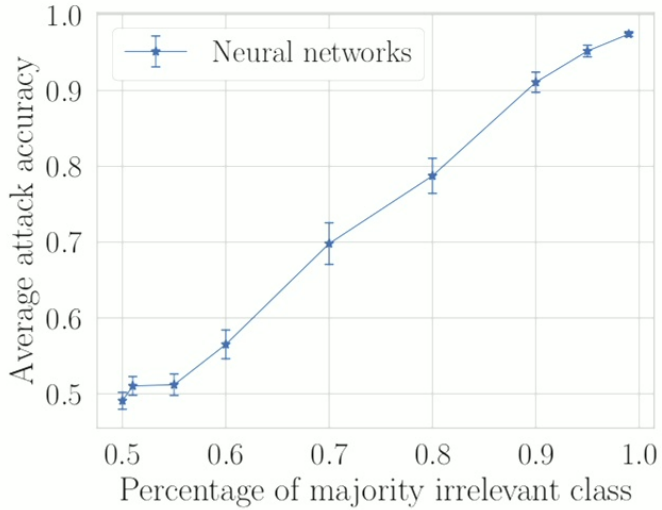


https://global.health

A sneak peek of the results

Over-powerful adversary

Old attack: logistic regression ($\#_{\text{params}} = 615$)

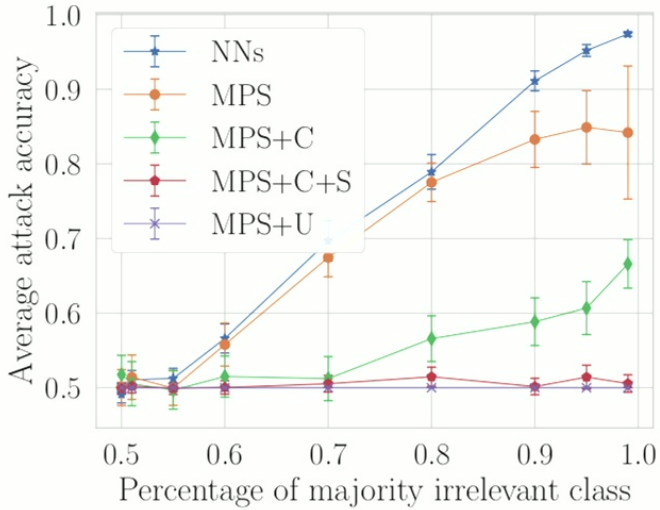


A sneak peek of the results

Over-powerful adversary

Old attack: logistic regression ($\#_{\text{params}} = 615$)

New attack: neural network ($d_{\text{in}} = 40$, $d_{\text{out}} = 2$, $\#_{\text{layers}} = 6$, $\#_{\text{params}} = 1588$)



How should a good model work?

A good model should extract from the training data whatever is necessary to perform the target task well, *and no more*.

$$\downarrow$$

Knowing the model's parameters should not be more informative about the training data than a black-box access.

A simple solution

If the parameters of a model contain undesired information on the training dataset, find an alternative parameterization of it (and don't use the training dataset for this).

$$\text{Invariance under reparameterizations} = \text{gauge symmetries}$$

Gauge symmetries for privacy

Theorem (Informal version)

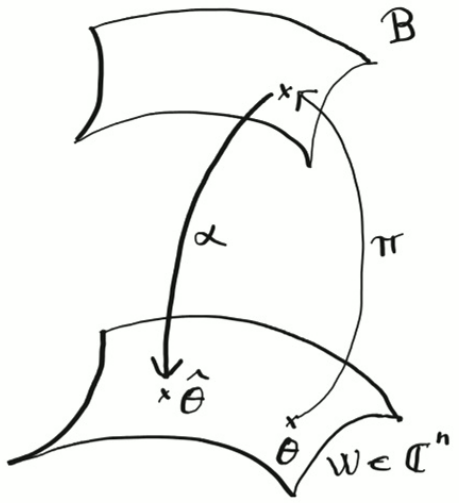
If:

– The sets of parameters that leave a model invariant can be characterized,

– There exists a way to assign parameters to a black box (e.g., training a neural network in queries),

Then:

The evaluation of every function applied to a representative of the set coincides with the evaluation of the induced function in the corresponding black box.



Gauge symmetries for privacy

Theorem (Informal version)

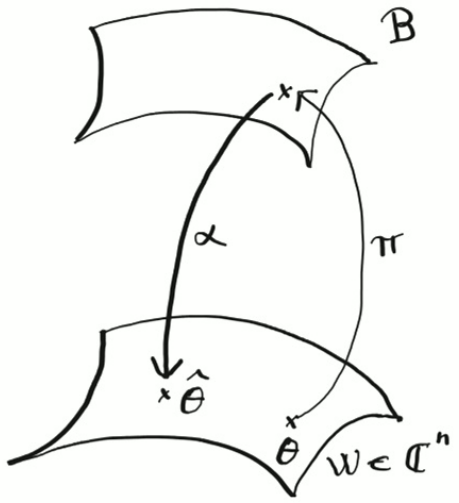
If:

– The sets of parameters that leave a model invariant can be characterized,

– There exists a way to assign parameters to a black box (e.g., training a neural network in queries),

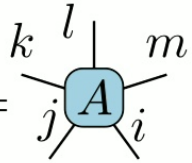
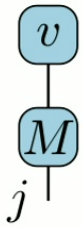
Then:

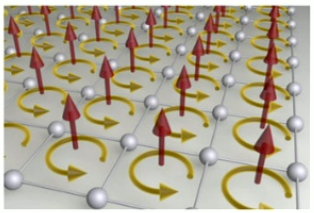
The evaluation of every function applied to a representative of the set coincides with the evaluation of the induced function in the corresponding black box.



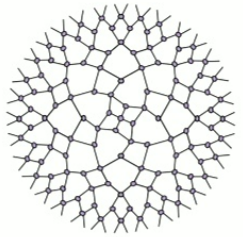
If some information cannot be accessed from queries, neither it can be accessed from the parameters $\hat{\theta}$.

ML architectures with gauge symmetries: tensor networks

$A_{ijklm} =$  $, \sum_i M_{ij} v_i =$ 



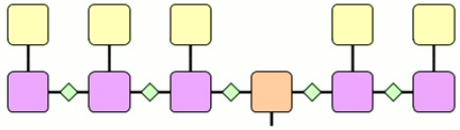
$(\mathbb{C}^d)^N$



poly(N)

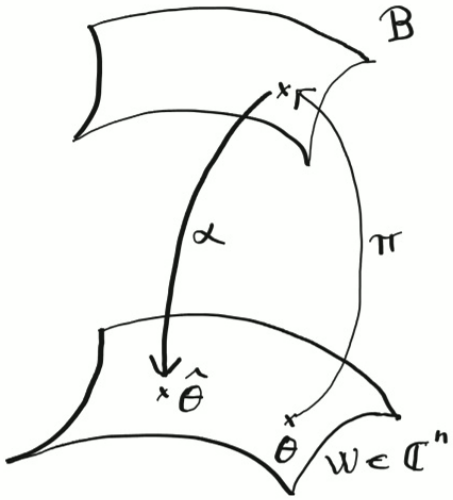
Matrix Product States (MPS)

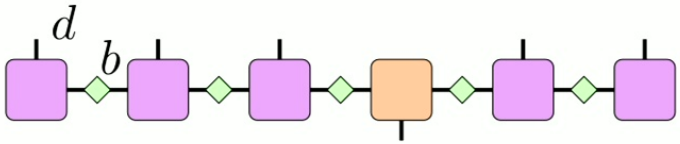
$x = (x_1, \ldots, x_N) \to \Phi(x) = \phi_1(x_1) \otimes \cdots \otimes \phi_N(x_N) =$ 

$f_\theta(x) = \Phi(x) \cdot M_\theta = \sum_{\{s\}} \phi_1^{s_1} \cdots \phi_N^{s_N} \sum_{\{\alpha\}} [A_1]_{s_1}^{\alpha_1} \cdots [A_N]_{s_N}^{\alpha_N} =$ 

First uses in ML in 2016 (arXiv:1605.05775), now competing with (arXiv:1806.05964) and surpassing (arXiv:2006.02516) traditional architectures in specific scenarios.

MPS for privacy-preserving ML





$\mathcal{W} = \mathbb{C}^{Nb^2d}$, $\mathcal{B} \in \mathbb{C}^{d^N}$

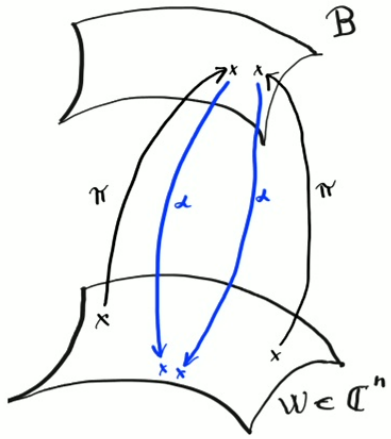
$\pi(\{A\}) = \sum_{\{\alpha\}} [A_1]_{s_1}^{\alpha_1} [A_2]_{s_2}^{\alpha_1,\alpha_2} \cdots [A_N]_{s_N}^{\alpha_N}$

$[\tilde{A}_i]_s = X_i^{-1} [A_i]_s X_{i+1} \Rightarrow \pi(\{A\}) = \pi(\{\tilde{A}\})$

$\alpha(W) = \{\hat{A}_1, \hat{A}_2, \cdots, \hat{A}_N\}$ via SVD

Not the end of the story

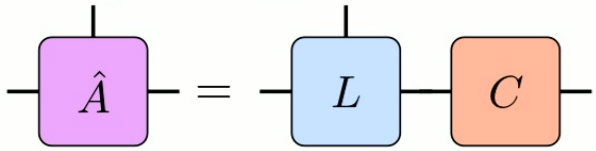
If there is information stored about the training set (useful for the task, useless but present because of imperfect training, ...), this could be easier to access by looking at the parameters than by input-output queries.

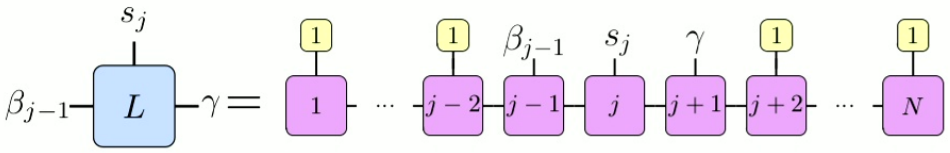


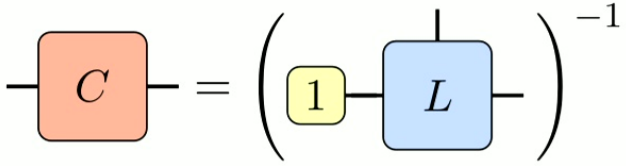
Theorem (Informal version)

There is a canonical form for the set of MPS architectures so that every white-box attack to such canonical-form set of parameters is "as good" (in terms of the attack accuracy and its regularity as a function) as an attack to the black-box representation.

Proof: A globally smooth canonical form for MPS







Holomorphic (smooth)
as long as L are invertible.

First univocal canonical form for MPS!

(Anonymous): Anything you discover is already known to Russian mathematicians

3.1. Skeleton decomposition in the exact case

Let us recall what the skeleton decomposition is. If a $m \times n$ matrix A has rank r, then it can be represented as

$$A = C\widehat{A}^{-1}R, \tag{9}$$

where $C = A(:, \mathcal{J})$ are some r columns of A, $R = A(\mathcal{I}, :)$ are some r rows of A and

$$\widehat{A} = A(\mathcal{I}, \mathcal{J})$$

[1] Oseledets and Tyrtyshnikov, Linear Algebra Appl. 432, 70–88 (2010)

Summary so far

Theorem 1: If the sets of parameters that leave a model invariant can be characterized, then each white-box attack performed on a representative of the set is equally performing (in terms of accuracy) as an attack in the corresponding black box.

In architectures with gauge symmetries, access to the canonical-form model $\leftrightarrow$ black-box access.

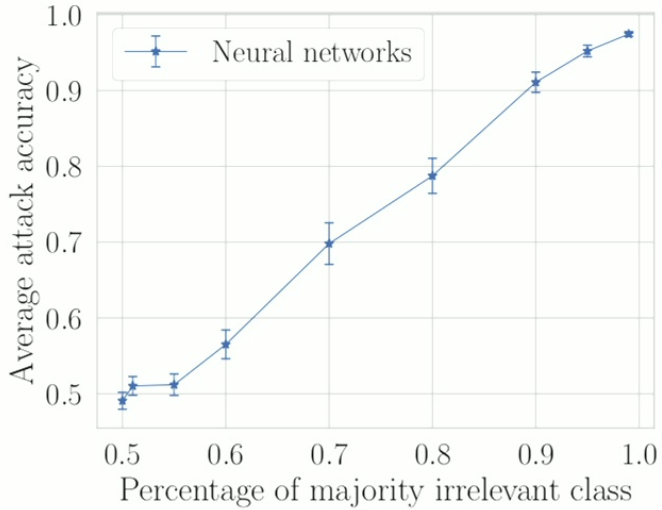
Theorem 2: There is a canonical form for the set of MPS architectures so that every white-box attack to such canonical-form set of parameters is "as good" (in terms of the attack accuracy and its regularity as a function) as an attack to the black-box representation.

For MPS architectures, if two models are close as black boxes their parameters are also close.

Privacy protection in practice

<u>Over-powerful adversary</u>

Old attack: logistic regression ($\#_{\text{params}} = 615$)

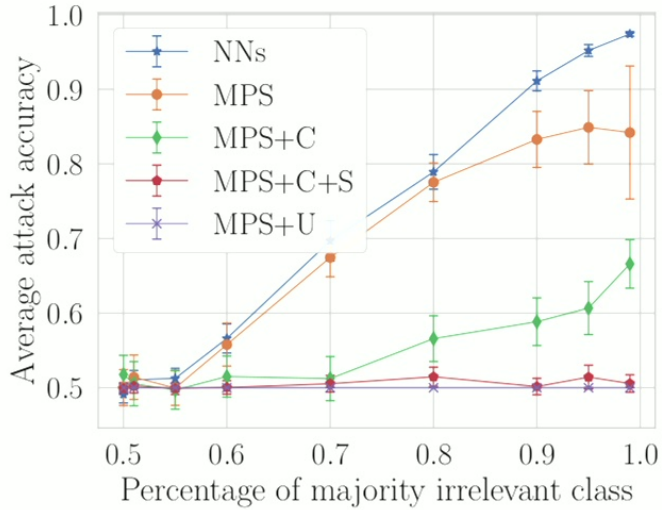


Privacy protection in practice

Over-powerful adversary

Old attack: logistic regression ($\#_{\text{params}} = 615$)

New attack: neural network ($d_{\text{in}} = 40$, $d_{\text{out}} = 2$, $\#_{\text{layers}} = 6$, $\#_{\text{params}} = 1588$)



Conclusions

– Result 0: Neural networks are vulnerable to privacy leaks that involve global properties of the training set.

  - Relations to DP (applicability, efficiency...).

– Result 1: Gauge symmetry provides a notion of privacy.

  - Find reparametrizations for standard network architectures.

– Result 2: Tensor networks provide favourable playground.

  - Privacy may not be at odds with performance.
  - Use architectures with canonical forms for ML, more motivation to find new ones.
  - Find canonical forms with desired properties.

Tensorkrowch: Create your TNML models in Pytorch

```python
import tensorkrowch as tk

# Create TensorNetwork
net = tk.TensorNetwork(name='my_net')

# Create Nodes in ``net``
node1 = tk.randn(shape=(5, 3),
                 axes_names=('input', 'right'),
                 network=net)
node2 = tk.randn(shape=(3, 5),
                 axes_names=('left', 'input'),
                 network=net)

# Connect nodes
node1['right'] ^ node2['left']

# Contract nodes
node3 = node1 @ node2
print(node3.shape)
```

```
torch.Size([5, 5])
```

```python
import torch

net = MyNet()

data = torch.randn(2, 100, 5)  # n_features x batch_size x feature_size
result = net(data)

print(result.shape)  # batch_size

# ``node1`` and ``node2`` have gradients, they can be "learned"
result.mean().backward()
```

```python
import torch.nn as nn

class MyModel(nn.Module):

    def __init__(self):
        super().__init__()

        self.mps = tk.MPSLayer(n_sites=10,
                               d_phys=5,
                               n_labels=20,
                               d_bond=10)
        self.linear = nn.Linear(20, 1)

    def forward(self, x):
        x = self.mps(x)
        x = self.linear(x)
        return x
```

José Pareja (ICMAT)

https://joserapa98.github.io/tensorkrowch

J. R. Pareja Monturiol, D. Pérez-García, and APK, arXiv:2306.08595

Thank you for your attention

Questions? Comments?

2202.12319 Private TNML
2306.08595 TNML on Pytorch

physics@alexpozas.com

apozas/private-tn
joserapa98/tensorkrowch

UNIVERSITÉ DE GENÈVE

CONSTRUCTOR INSTITUTE