Efficient Evaluation of Activation Functions over Encrypted Data

Patricia Thaine, Prof. Sergey Gorbunov, Prof. Gerald Penn

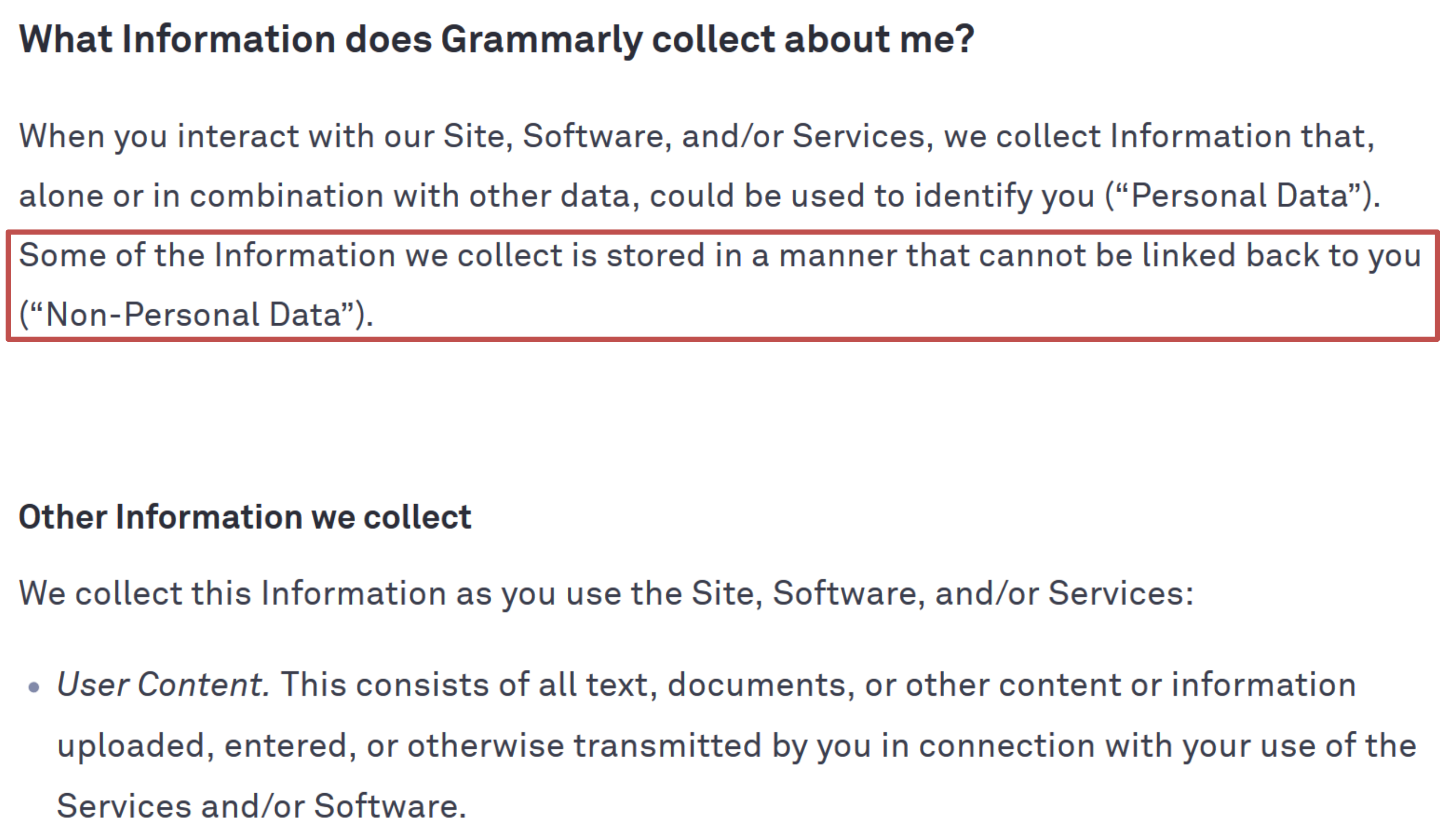
Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

WHY PRIVACY-PRESERVING ML?

The utility of our personally identifiable information is pervasive and **we don't know who it's being shared with!**

WHY PRIVACY-PRESERVING ML?



Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

WHY PRIVACY-PRESERVING ML?

Dropbox

We need your permission to do things like hosting Your Stuff, backing it up, and sharing it when you ask us to. Our Services also provide you with features like photo thumbnails, document previews, commenting, easy sorting, editing, sharing, and searching. These and other features may require our systems to access, store, and scan Your Stuff. You give us permission to do those things, and this permission extends to our affiliates and trusted third parties we work with.

WHY PRIVACY-PRESERVING ML?

With over 2.6 billion records breached in 2017 alone (76% due to accidental loss, 23% due to malicious outsiders) [1] and a growing shortage of cybersecurity professionals:
more data privacy = more data security

[1] https://breachlevelindex.com/assets/Breach-Level-Index-Report-2017-Gemalto.pdf

SOME ML TASKS THAT USE SENSITIVE DATA

Gait Detection

Fingerprint Recognition

Facial Recognition

Authorship Recognition

Machine Translation

Named Entity Recognition

Recommendation Systems

Question Answering

Automatic Speech Recognition

Text-to-Speech

Speaker Recognition

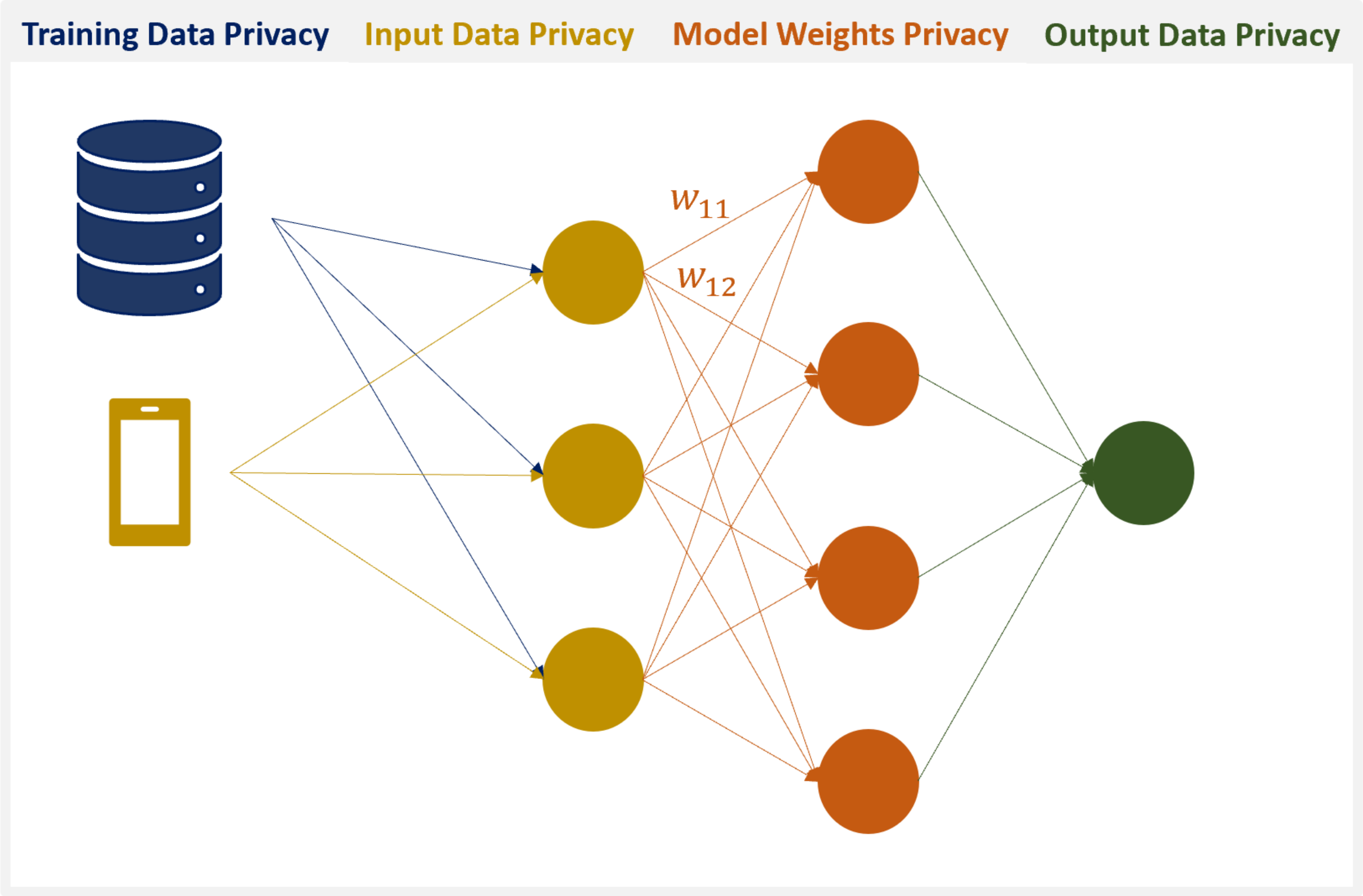
Speaker Profiling

Disease Prediction

Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

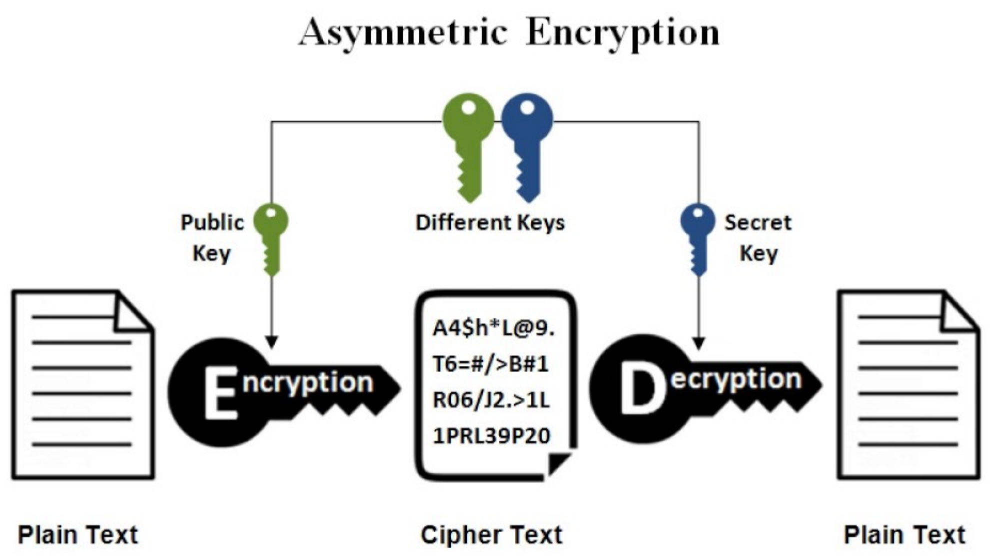
WHAT WOULD PERFECTLY PRIVACY-PRESERVING ML LOOK LIKE?

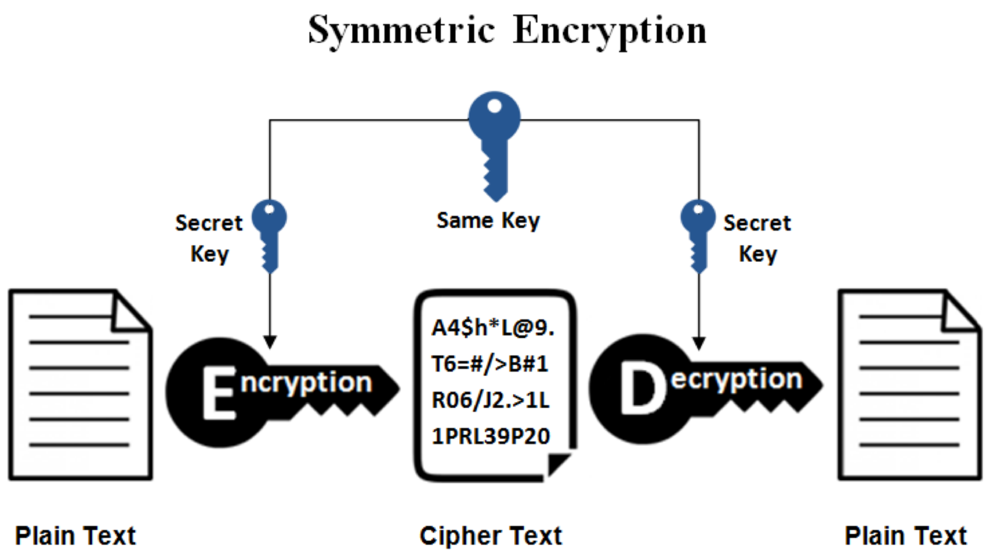


Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

CRYPTOGRAPHY





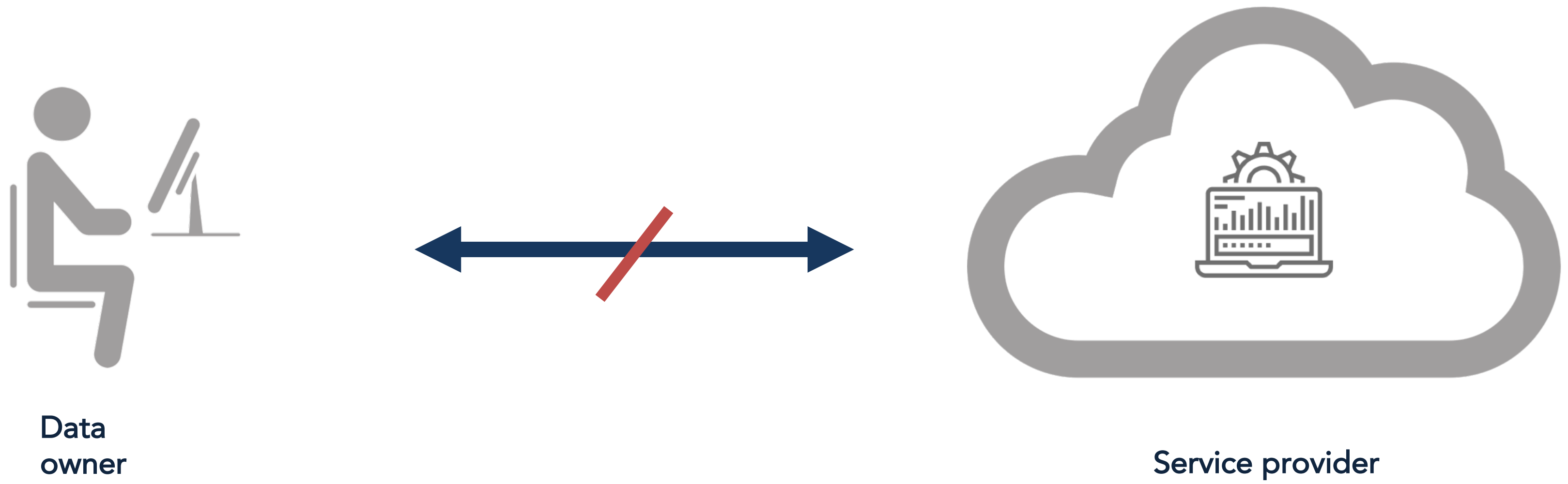
Sources: https://i.ytimg.com/vi/-1FInW1HCbw/maxresdefault.jpg https://www.ssl2buy.com/wiki/wp-content/uploads/2015/12/Symmetric-Encryption.png

Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

SECURE TWO-PARTY COMPUTATION

Suppose we were able to use 2PC to provide input and output data privacy …

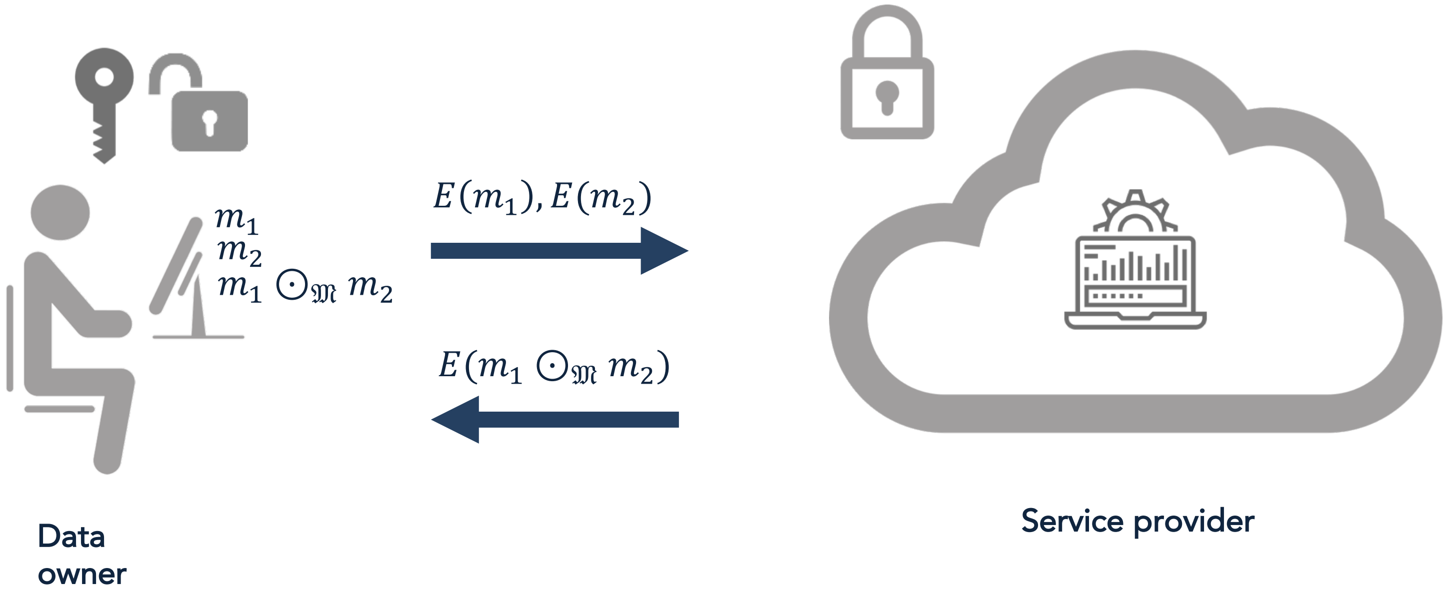


Limitations:

- Could incur very high communication costs
- Data owner could have low computational capacity
- Data owner could be offline

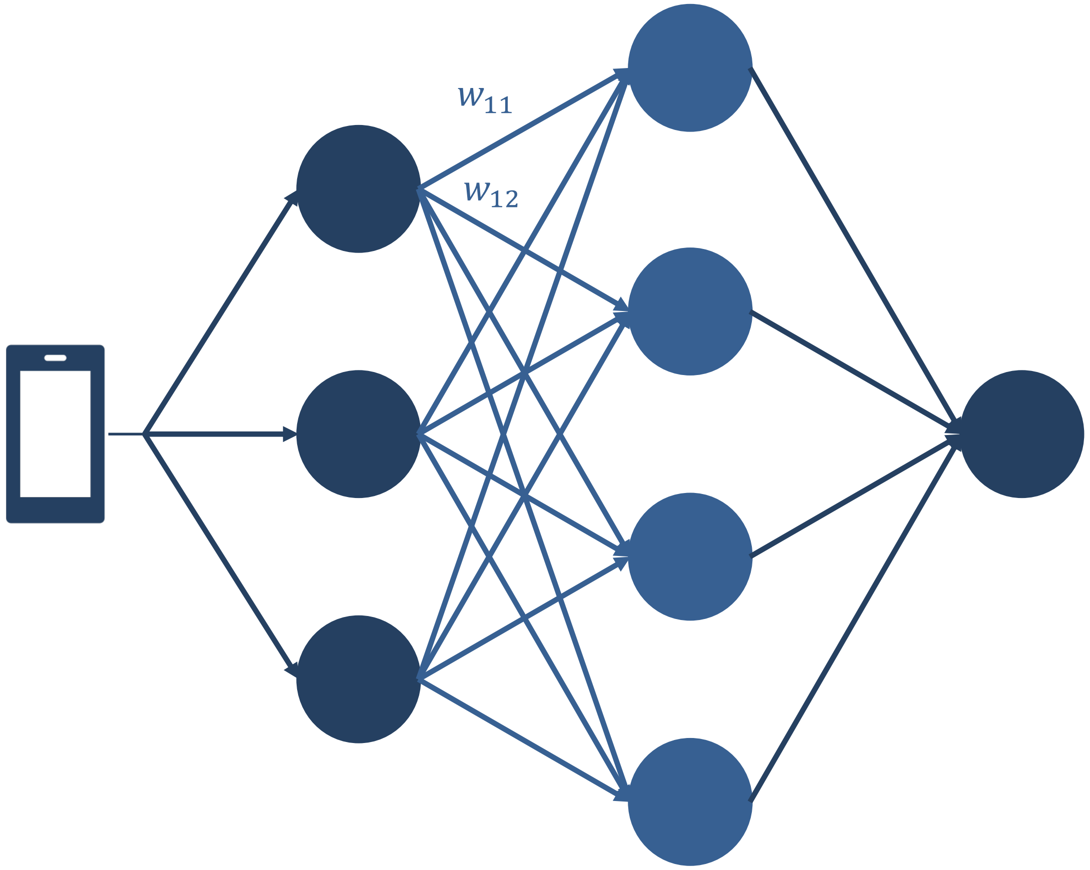
HOMOMORPHIC ENCRYPTION

$$\forall m_1, m_2 \in \mathfrak{M}, E(m_1 \odot_{\mathfrak{M}} m_2) \leftarrow E(m_1) \odot_{\mathfrak{M}} E(m_2)$$



HOMOMORPHIC ENCRYPTION

1. training data privacy;
2. input data privacy;
3. model weight privacy;
4. output data privacy.



Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

WHY HOMOMORPHIC ENCRYPTION?

Semantically secure probabilistic encryption: "for any function f and any plaintext m, and with only polynomial resources [...], the probability to guess $f(m)$ (knowing f but not m) does not increase if the adversary knows a ciphertext corresponding to m" (Fontaine and Garland 2007).

HOMOMORPHIC ENCRYPTION IN PRACTICE

Easy Operations: linear and polynomial

Difficult Operations: non-polynomial

DEALING WITH NON-POLYNOMIAL EQUATIONS IN PRIVATE DEEP LEARNING

PRIOR WORK

- $f(x) = x^2$ used as an activation function instead of ReLU (Gilad-Bachrach et al., 2016).
- Distant polynomial approximation of sigmoid function used for training a neural network on encrypted data (Hesamifard et al., 2016).

PRIVATE DL COPING METHOD I

$f(x) = x^2$ used as an activation function instead of ReLU in CryptoNets. No alternative proposed for sigmoid. 99% accuracy on MNIST OCR (Gilad-Bachrach et al., 2016).

Table 1. Breakdown of the time it takes to apply CryptoNets to the MNIST network

Layer	**Description**	**Time to compute**
Convolution layer	Weighted sums layer with windows of size 5×5, stride size of 2. From each window, 5 different maps are computed and a padding is added to the upper side and left side of each image.	30 seconds
1^{st} square layer	Squares each of the 835 outputs of the convolution layer	81 seconds
Pool layer	Weighted sum layer that generates 100 outputs from the 835 outputs of the 1^{st} square layer	127 seconds
2^{nd} square layer	Squares each of the 100 outputs of the pool layer	10 seconds
Output layer	Weighted sum that generates 10 outputs (corresponding to the 10 digits) from the 100 outputs of the 2^{nd} square layer	1.6 seconds

Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

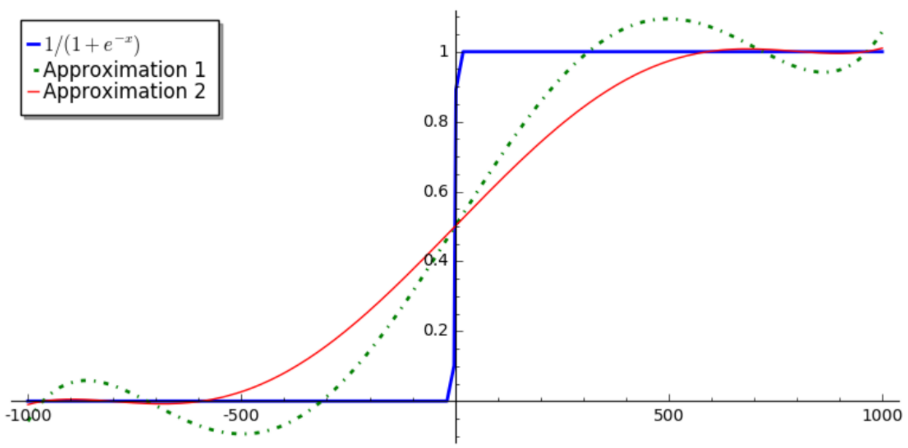
PRIVATE DL COPING METHOD II

Approximation of sigmoid function used for training a neural network on encrypted data (Hesamifard et al., 2016/2017).

Table 1: Polynomial approximation of sigmoid function on Interval $[-10^3, 10^3]$

$$p_1(x) = (2.069e-15) * x^5 + \cdots + 0.001 * x + 0.499$$

$$p_2(x) = (6.653e-16) * x^5 + \cdots + 0.001 * x + 0.500$$



CONTRIBUTION

We show how to represent the value of any function over a defined and bounded interval, given encrypted input data, without needing to decrypt any intermediate values before obtaining the function's output.

~~PRIVACY-PRESERVING MACHINE LEARNING~~
PRIVACY-PRESERVING NUMERICAL COMPUTATION

Computer Science
UNIVERSITY OF TORONTO

University of
Waterloo

OUR SETUP AND NOTATION

We use the RLWE-based Brakerski/Fan-Vercauteren (B/FV) homomorphic encryption scheme.

We perform component-wise addition and component-wise multiplication in the encrypted domain.

We use E$(*)$ to denote that $*$ is an encrypted value.

We encode floating point numbers by multiplying them by 10^{ϕ} and rounding to the nearest integer, where ϕ is our desired level of precision.

HOMOMORPHIC ENCRYPTION OVERVIEW

Component-wise vs. Polynomial Operations

Addition	Multiplication	Addition	Multiplication
$0x^4 + 4x^3 + 6x^2 + 2x + 5$ $+ \underline{1x^4 + 6x^3 + 3x^2 + 5x + 2}$ $1x^4 + 10x^3 + 9x^2 + 7x + 7$	$0x^4 + 4x^3 + 6x^2 + 2x + 5$ $* \underline{1x^4 + 6x^3 + 3x^2 + 5x + 2}$ $0x^4 + 24x^3 + 18x^2 + 10x + 10$	$0x^4 + 4x^3 + 6x^2 + 2x + 5$ $+ \underline{1x^4 + 6x^3 + 3x^2 + 5x + 2}$ $1x^4 + 10x^3 + 9x^2 + 7x + 7$	$0x^4 + 4x^3 + 6x^2 + 2x + 5$ $* \underline{1x^4 + 6x^3 + 3x^2 + 5x + 2}$ $4x^7+30x^6+50x^5+55x^4+74x^3+37x^2$ $+29x+10$

Option #1

Option #2

Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

SECURITY, INTEGRITY, AND CORRECTNESS

1) No information about the inputs provided by the client is revealed to even a malicious server.

2) Assuming the server is semi-honest, no information about the inputs is revealed, and the client learns the correct results of its desired computations.

Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

EFFICIENT TABLE LOOKUP

Input: an encrypted number $E(x_i)$, a function f, and a range of values (e.g., 1 to 8) with a step between those values (e.g., 1).

Output: $y_i = f(x_i)$

Step 1: create a vector of indices, I, from the input range, a vector of the results of f applied to each of these indices plus 1 denoted by $f(I)$, and a vector, X, which has $\mathrm{E}(x_i)$ as a repeated value. Say, $x_i = 4$.

$$I = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{bmatrix}, f(I) = \begin{bmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \\ f(5) \\ f(6) \\ f(7) \\ f(8) \\ 0 \end{bmatrix}, X = \begin{bmatrix} E(4) \\ E(4) \\ E(4) \\ E(4) \\ E(4) \\ E(4) \\ E(4) \\ E(4) \\ E(4) \end{bmatrix}$$

Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

EFFICIENT TABLE LOOKUP

Step 2: subtract.

$$\begin{bmatrix} E(4) \\ E(4) \\ E(4) \\ E(4) \\ E(4) \\ E(4) \\ E(4) \\ E(4) \\ E(4) \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{bmatrix} = \begin{bmatrix} E(4) \\ E(3) \\ E(2) \\ E(1) \\ E(0) \\ E(-1) \\ E(-2) \\ E(-3) \\ E(-4) \end{bmatrix}$$

EFFICIENT TABLE LOOKUP

Step 3: rotate by one and multiply.

$$\begin{bmatrix} E(4) \\ E(3) \\ E(2) \\ E(1) \\ E(0) \\ E(-1) \\ E(-2) \\ E(-3) \\ E(-4) \end{bmatrix} \times \begin{bmatrix} E(-4) \\ E(4) \\ E(3) \\ E(2) \\ E(1) \\ E(0) \\ E(-1) \\ E(-2) \\ E(-3) \end{bmatrix} = \begin{bmatrix} E(-16) \\ E(12) \\ E(6) \\ E(2) \\ E(0) \\ E(0) \\ E(2) \\ E(6) \\ E(12) \end{bmatrix}$$

EFFICIENT TABLE LOOKUP

Step 4: rotate by two and multiply.

$$\begin{bmatrix} E(-16) \\ E(12) \\ E(6) \\ E(2) \\ E(0) \\ E(0) \\ E(2) \\ E(6) \\ E(12) \end{bmatrix} \times \begin{bmatrix} E(6) \\ E(12) \\ E(-16) \\ E(12) \\ E(6) \\ E(2) \\ E(0) \\ E(0) \\ E(2) \end{bmatrix} = \begin{bmatrix} E(-96) \\ E(144) \\ E(-96) \\ E(24) \\ E(0) \\ E(0) \\ E(0) \\ E(0) \\ E(24) \end{bmatrix}$$

EFFICIENT TABLE LOOKUP

Step 5: rotate by four and multiply.

$$\begin{bmatrix} E(-96) \\ E(144) \\ E(-96) \\ E(24) \\ E(0) \\ E(0) \\ E(0) \\ E(0) \\ E(24) \end{bmatrix} \times \begin{bmatrix} E(0) \\ E(0) \\ E(0) \\ E(24) \\ E(-96) \\ E(144) \\ E(-96) \\ E(24) \\ E(0) \end{bmatrix} = \begin{bmatrix} E(0) \\ E(0) \\ E(0) \\ E(576) \\ E(0) \\ E(0) \\ E(0) \\ E(0) \\ E(0) \end{bmatrix}$$

Uh oh!

Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

EFFICIENT TABLE LOOKUP

Step 6 (preamble): We can…

- Simply keep track of a denominator?

  Simple in the short term, potentially problematic in the long term.

Or…

- Exploit the fact that RLWE-based cryptosystems use plaintext moduli!

  E.g., $[0x^4 + 4x^3 + 6x^2 + 2x + 5]_7$

  $+ [1x^4 + 6x^3 + 3x^2 + 5x + 2]_7$

  $= [1x^4 + 3x^3 + 2x^2 + 0x + 0]_7$

EFFICIENT TABLE LOOKUP

Step 6 (preamble): Since we know I, as well as every possible value that x_i can be, and the plaintext modulus p, we can pre-compute the following vectors (say $\mathrm{p} = 65537$):

$$\begin{bmatrix} 7711 \\ 5780 \\ 53977 \\ \mathbf{14450} \\ 53977 \\ 5780 \\ 7711 \\ 56381 \\ 0 \end{bmatrix} \times \begin{bmatrix} -5040 \\ 1440 \\ -720 \\ \mathbf{576} \\ -720 \\ 1440 \\ -5040 \\ 40320 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \pmod{65537} \\ 1 \pmod{65537} \\ 1 \pmod{65537} \\ \mathbf{1} \ (\mathbf{mod}\ \mathbf{65537}) \\ 1 \pmod{65537} \\ 1 \pmod{65537} \\ 1 \pmod{65537} \\ 1 \pmod{65537} \\ 0 \end{bmatrix}$$

EFFICIENT TABLE LOOKUP

Step 6:

$$\begin{bmatrix} 7711 \\ 5780 \\ 53977 \\ \mathbf{14450} \\ 53977 \\ 5780 \\ 7711 \\ 56381 \\ 0 \end{bmatrix} \times \begin{bmatrix} E(0) \\ E(0) \\ E(0) \\ E(576) \\ E(0) \\ E(0) \\ E(0) \\ E(0) \\ E(0) \end{bmatrix} = \begin{bmatrix} E(0) \\ E(0) \\ E(0) \\ E(1) \\ E(0) \\ E(0) \\ E(0) \\ E(0) \\ E(0) \end{bmatrix}$$

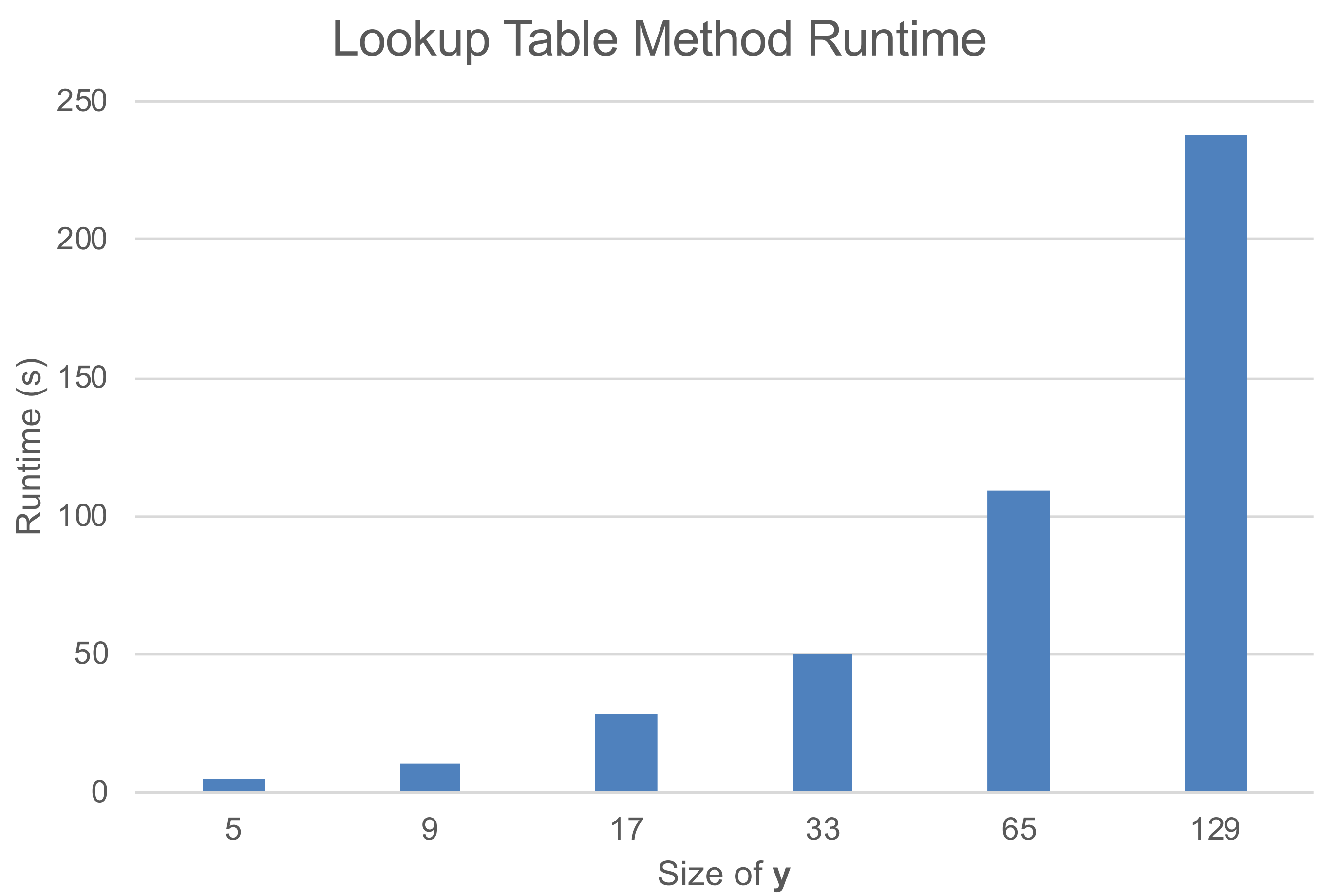
EFFICIENT TABLE LOOKUP

Step 7: Solved in $\log(n) + 2$ multiplications!

$$\begin{bmatrix} E(0) \\ E(0) \\ E(0) \\ E(1) \\ E(0) \\ E(0) \\ E(0) \\ E(0) \\ E(0) \end{bmatrix} \cdot \begin{bmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \\ f(5) \\ f(6) \\ f(7) \\ f(8) \\ 0 \end{bmatrix} = E(f(4))$$

EFFICIENT TABLE LOOKUP

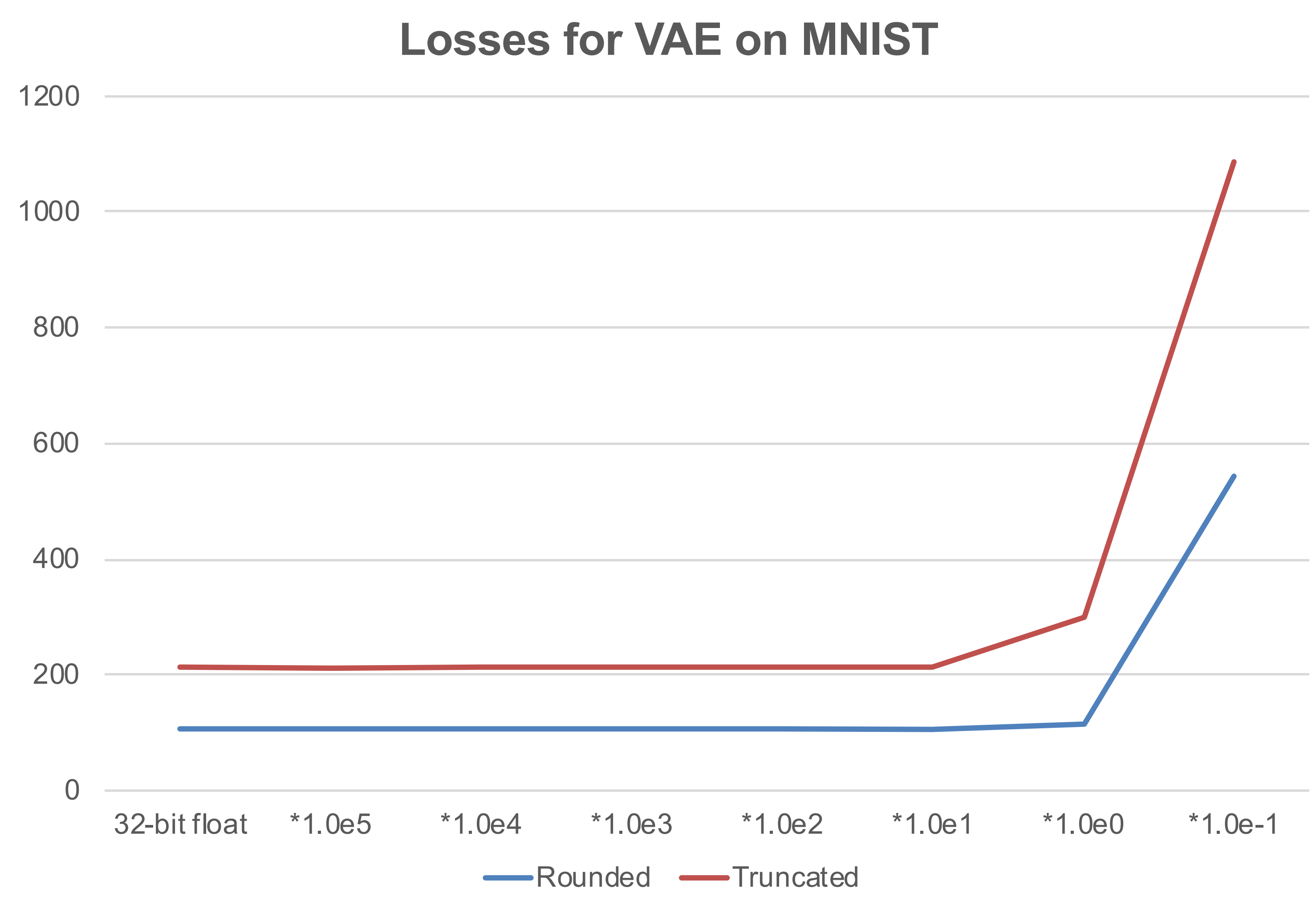
Results for over a 256-bit security level, using an Intel Core i-7-8650U CPU @1.90GHz and 16GB RAM. Runtime increments linearly with the size of the lookup table.



Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

EXPERIMENTS: VARIATIONAL AUTOENCODER (VAE)

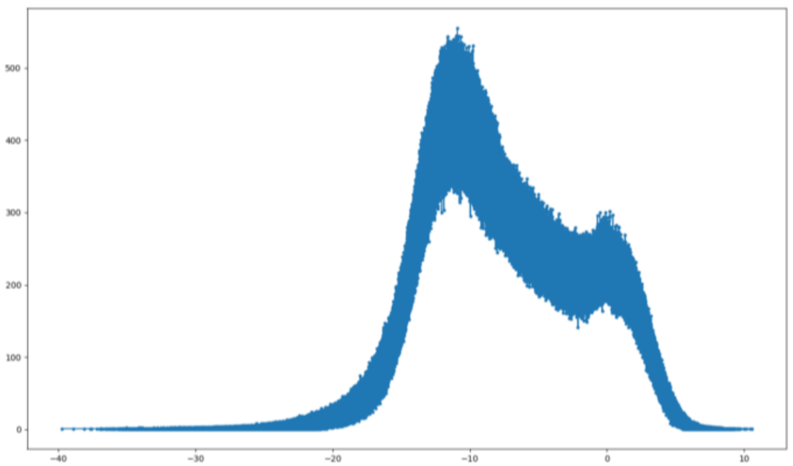


- Replacing the 2 ReLU and 1 sigmoid with our approximation method.
- Loss minimized at $\phi = 1$ (truncation method).
- Loss at $\phi = 0$ (rounding method) still reasonable.

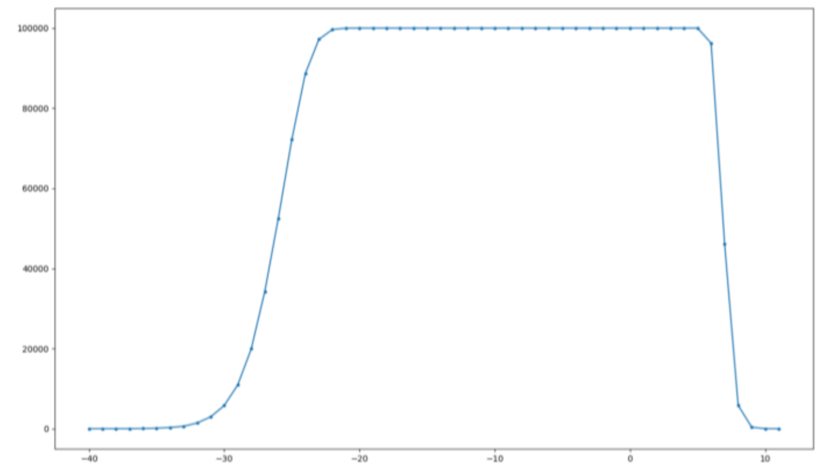
Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

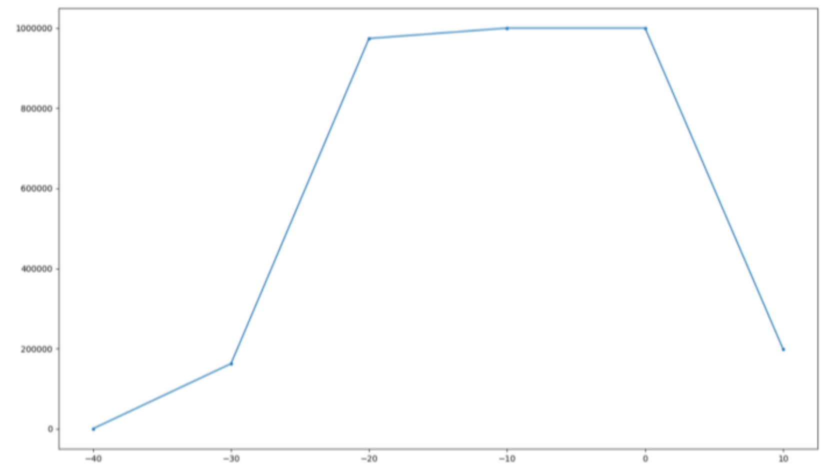
EXPERIMENTS: VARIATIONAL AUTOENCODER (VAE)



(a) $\phi = 5$



(b) $\phi = 0$



(c) $\phi = -1$

- Aggregate number of distinct values over 10 epochs input into VAE's sigmoid function.
- x-axis: input values; y-axis: quantity of inputs with those values.
- (a) 549301760 many distinct values; (b) 52; (c) 6.
- We only need a lookup table of size 65 for this sigmoid function!

Computer Science
UNIVERSITY OF TORONTO

University of
Waterloo

EXPERIMENTS: MNIST IMAGE CLASSIFICATION

	original	*1.0e5	*1.0e4	*1.0e3	*1.0e2	*1.0e1	*1.0e0	*1.0e-1
Loss (Rounded)	0.0524	0.0531	0.0535	0.0542	0.0541	0.0534	0.0642	2.301
# Correct (Rounded)	9839	9835	9838	9836	9832	9841	9807	1135
Loss (Truncated)	0.0524	0.0526	0.0535	0.0548	0.0526	0.0542	2.3011	2.3011
# Correct (Truncated)	9839	9838	9834	9828	9836	9829	1135	1135

Resulting losses and number of correct classifications of 10000 test set images from MNIST with the inputs to its three ReLU activation functions approximated at various precisions.

TAKEAWAYS

- Using HE for ML is less of an ML problem and more of a NA problem.
- We *can* protect users' private data while continuing to use them for ML in general.
- When deciding how to implement a neural network using homomorphic encryption, we need a very clear understanding of the problem we are solving.

Computer Science
UNIVERSITY OF TORONTO

University of Waterloo

Thank you!

@PrivateNLP

https://medium.com/privacy-preserving-natural-language-processing

Computer Science
UNIVERSITY OF TORONTO

University of
Waterloo

REFERENCES

Brakerski, Zvika. “Fully homomorphic encryption without modulus switching from classical GapSVP.” In Advances in cryptology 2012, pp. 868-886. Springer, Berlin, Heidelberg, 2012.

Fan, Junfeng, and Frederik Vercauteren. “Somewhat Practical Fully Homomorphic Encryption.” IACR Cryptology ePrint Archive 2012 (2012): 144.

Fontaine, Caroline, and Fabien Galand. ”A survey of homomorphic encryption for nonspecialists.” EURASIP Journal on Information Security 1 (2009): 41-50.

Gilad-Bachrach, Ran, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. “Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy.” In International Conference on Machine Learning, pp. 201-210. 2016.

Hesamifard, Ehsan, Hassan Takabi, and Mehdi Ghasemi. “CryptoDL: Towards Deep Learning over Encrypted Data.” In Annual Computer Security Applications Conference (ACSAC 2016), Los Angeles, California, USA. 2016.

Hesamifard, Ehsan, Hassan Takabi, Mehdi Ghasemi, and Rebecca N. Wright. “Privacy-preserving machine learning as a service.” Proceedings on Privacy Enhancing Technologies 2018, no. 3 (2018): 123-142.

Kingma, Durk P., Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. “Semi-supervised learning with deep generative models.” In Advances in neural information processing systems, pp. 3581-3589. 2014.