GDAŃSK UNIVERSITY
OF TECHNOLOGY

Privacy-Preserving Machine Learning Inference

Stanislaw Baranski

April 8, 2025

GDAŃSK UNIVERSITY OF TECHNOLOGY

Outline

- Client delegates the ML service to service provider (Server)
- Separation of specialization
- Cost reductions

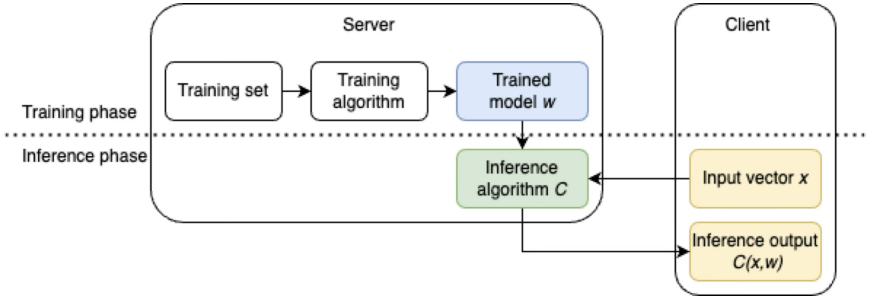
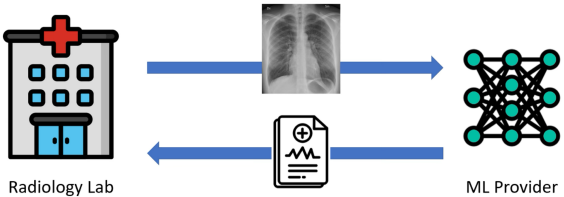


Fig 1: Typical MLaaS setup

Use Cases - Medical prognosis using ML

Problems:

- The ML Provider cannot share the model as it may be proprietary
- The laboratory can not send input data to the ML Provide because of legal prohibitions or complex agreements.
- Privacy risks

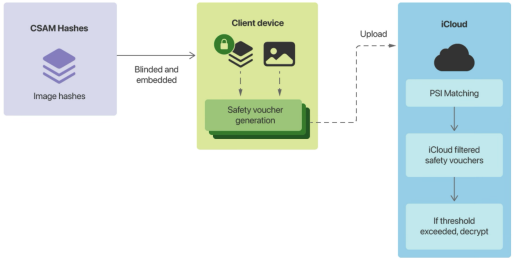


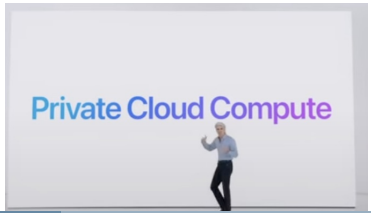
Is it possible to do this computation without the radiology lab ever sharing the patient's sensitive data and the ML provider sharing its proprietary model?

Use Cases - Private set intersection, Apple CSA

CSAM Detection enables Apple to accurately identify and report iCloud users who store known Child Sexual Abuse Material (CSAM) in their iCloud Photos accounts. Apple servers flag accounts exceeding a threshold number of images that match a known database of CSAM image hashes so that Apple can provide relevant information to the National Center for Missing and Exploited Children (NCMEC). This process is secure, and is expressly designed to preserve user privacy. [1], [2]

- Apple CSAM Detection
- Password Monitoring
- Communication Safety

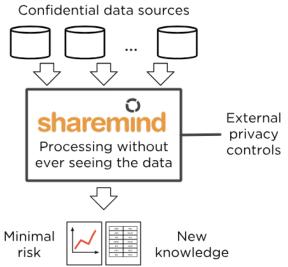




Use Cases - Estonian students study I

From [3]

Estonian students study. *In Estonia, a country with arguably the most advanced e-government and technology awareness, alarms were raised about graduation rates of IT students. Surprisingly, in* **2012, nearly 43% of IT studentsen rolled in the previous five years had failed to graduate**. *One potential explanation considered was that the* **IT industry was hiring too aggressively, luring students away from completing their studies.** *The Estonian Association of Information and Communication Technology wanted to* **investigate by mining education and tax records to see if there was a correlation.** *However,* **privacy legislation prevented data sharing across the Ministry of Education and the Tax Board.** *In fact, k-anonymity-based sharing was allowed, but it would have resulted in low-quality analysis, since many students would not have had sufficiently large groups of peerswith similar qualities.* **MPC provided a solution**, *facilitated by the Estonian company Cybernetica using their Sharemind framework (Bogdanov et al., 2008a). The data analysis was done as a three-party computation, with servers representing the Estonian Information System's Authority,the Ministry of Finance, and Cybernetica. The study, reported in Cybernetica (2015) and Bogdanov(2015), found that there was no correlation between working during studies and failure to graduate on time, but that more education was correlated with higher income.*



Other applications

- Secure machine learning
- privacy-preserving network security monitoring (Burkhart et al., 2010)
- privacy-preserving genomics (Wang et al., 2015a; Jagadeesh et al., 2017)
- Contact discovery (Li et al., 2013; De Cristofaro et al., 2013)
- Spam filtering on encrypted email (Gupta et al., 2017)
- Internet Voting, Credit score, everything which involves sensitive data...

The Rise of MLaaS and Privacy Concerns

- **Machine Learning as a Service (MLaaS):** Powerful models hosted in the cloud.
- Users send data for inference (predictions).
- **Problem:** Sensitive data exposure!
  - Healthcare (HIPAA, GDPR) [4]
  - Financial data
  - Personal images, text, etc.
- **Also:** Companies want to protect their proprietary models [5].
- **Without protection:** Inference attacks can leak input data or model details [6].

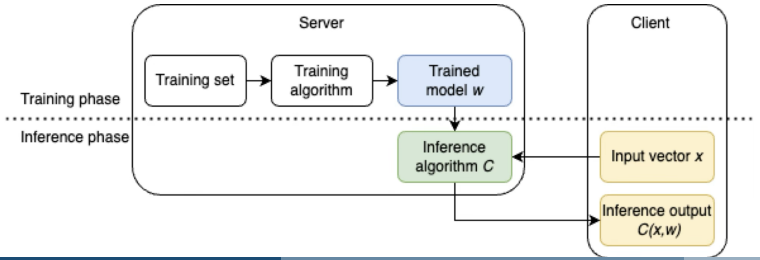


Fig 1: Typical MLaaS setup

What is Privacy-Preserving ML (PPML) Inferen

Goal: Compute the prediction $f(x, w)$ without revealing sensitive information.

Client (Data Owner)

- Has private input x (e.g., medical image, financial record).
- Wants the prediction $y = f(x, w)$.
- Privacy Need: Keep x secret from the Server.

Server (Model Owner)

- Has the ML model $f(\cdot)$ with parameters w.
- Computes the prediction for the client.
- Privacy Need (Optional): Keep model w secret from the Client (protect IP).

Key Challenge

How can the server compute $f(x, w)$ on data x it cannot see, potentially using a model w the client cannot see?

Solution: Privacy Preserving Machine Learning

Sometimes also called Oblivious Neural Networks.

The solution requires drawing knowledge from four fields of science

- Machine learning
- Computation theory,
- Digital systems theory,
- Cryptography,

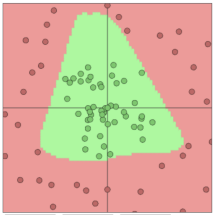
Neural Networks: Layers and Operations

- We focus on the **inference phase** (model is already trained).
- Neural networks are pipelines of layers.
- Each layer receives an input signal, transforms it, and generates an output signal, which becomes the input for the next layer. The first layer receives the input data x, and the output signal of the last layer is the prediction result $f(x, w)$.

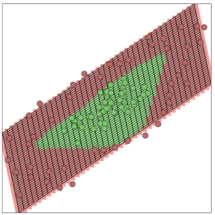
$$x \to f_1 \to a_1 \to ... \to f_n \to a_n \to y$$

- There are two types of transformations:
  - **Linear Transformation:** Matrix multiplication + addition ($y = Wx + b$). Includes convolutions.
  - **Nonlinear Transformation:** Activation functions (ReLU, Sigmoid, etc.) and Pooling.
- **Goal of Transformations:** Make data separable (classifiable).

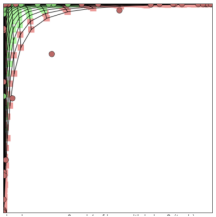
Visualizing layer transformations

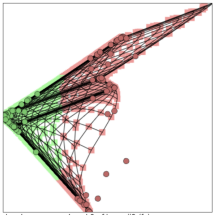


$y := W \cdot x + b$

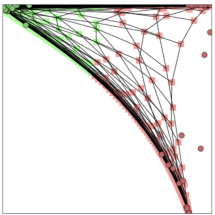


$y := \tanh(x)$

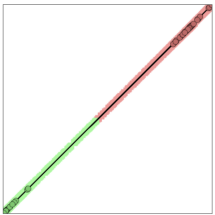




$y := W \cdot x + b$



$y := \tanh(x)$



$z := W \cdot x + b$

Key Operations for PPML

Recap: NN inference boils down to a sequence of:

- **Linear Ops:** Matrix Multiplication, Addition.
- **Non-Linear Ops:** Activation Functions (e.g., ReLU: $\max(0, x)$), Pooling (e.g., Max Pooling).

The PPML Connection

To make inference private, we need secure ways to perform these fundamental operations on hidden data/models!

Threat Model: Who Are We Protecting Against

- **Honest-but-Curious (Semi-Honest):** Standard assumption [7].
  - Client and Server follow the protocol correctly.
  - But... they might try to learn extra information from the messages they receive.
- *Malicious Model:* Stronger threat (parties might cheat). Requires more complex defenses (e.g., ZK Proofs), often too slow for now.
- **Privacy Goals Recap:**
  1. Protect **Input/Output Privacy**: Server doesn't learn client's x. Client learns only $f(x, w)$. (Main focus of crypto methods).
  2. Protect **Training Data Privacy**: Client queries shouldn't reveal info about the *original* training dataset [8]. (Where DP helps).

Approach 1: Homomorphic Encryption (HE)

Concept: Compute directly on encrypted data! [9]

1. Client encrypts input x with public key $pk \rightarrow Enc(x)$.
2. Client sends $Enc(x)$ to Server.
3. Server computes $Enc(f(x, w))$ using HE operations on $Enc(x)$ and potentially encrypted w.
4. Server sends $Enc(f(x, w))$ back to Client.
5. Client decrypts with secret key $sk \rightarrow f(x, w)$.

Pros:

- ✓ Strong crypto guarantees for data privacy.
- ✓ Low communication rounds (often just 1).
- ✓ No trusted third party needed.
- ✓ Supports batching (SIMD).

Cons:

- ✖ Very high computation cost (slow!).
- ✖ Large ciphertext sizes.
- ✖ Limited operations (non-linear are hard, need approximations like polynomials $\rightarrow$ accuracy loss).
- ✖ Noise growth limits computation depth (needs costly bootstrapping).

Example: CryptoNets (2016) showed feasibility on MNIST, but slow. [9]

Approach 2: Secure Multi-Party Computation (

Concept: Multiple parties jointly compute a function on their private inputs without revealing them [5]. **Common Techniques:**

- **Secret Sharing:** Input x and model w split into shares $[x_1], [x_2]$ and $[w_1], [w_2]$. Shares given to non-colluding servers (or client + server). Servers compute on shares. Linear ops are easy!
- **Garbled Circuits (GC):** Encrypts the logic of the function. Good for non-linear ops (like ReLU), often used with secret sharing.

Pros:

- ✓ Often faster computation than HE.
- ✓ Can handle non-linear functions well (using GC).
- ✓ Naturally protects both input and model.
- ✓ Many frameworks exist (CrypTen, TF Encrypted).

Cons:

- ✗ High communication cost (many rounds, large data transfer).
- ✗ Often requires multiple (>1) non-colluding servers for efficiency/simplicity.
- ✗ Can be complex to implement correctly.

Hybrid HE + MPC: Best of Both Worlds?

Idea: Combine HE and MPC to leverage their strengths [7]. **Common Strategy (e.g., Gazelle [10]):**

- Use HE or Additive Secret Sharing for **Linear Layers** (efficient additions/multiplications).
- Use Garbled Circuits (MPC) for **Non-Linear Layers** (like ReLU).

Why this works

Linear layers dominate compute time but are HE/Secret-Sharing friendly. Non-linear layers are bottlenecks for HE but cheaper with GC.

Results: Significant speedups compared to pure HE or pure MPC. Gazelle was 20x faster than prior MPC, 1000x faster than CryptoNets (HE) [10]. *Many state-of-the-art systems use hybrid approaches (e.g., Delphi [5]).*

Approach 3: Trusted Execution Environments (T

Concept: Use secure hardware enclaves (like Intel SGX, ARM TrustZone) [11].

1. Server runs inference code inside a protected hardware "bubble" (enclave).
2. Client uses *Remote Attestation* to verify the correct code is running in a genuine TEE.
3. Client encrypts data x, sends it to the enclave.
4. Enclave decrypts x, computes $f(x, w)$ on **plaintext** data inside the secure bubble.
5. Enclave encrypts result $f(x, w)$, sends back to Client.

Pros:

- ✓ Near-native performance (plaintext compute inside).
- ✓ Low latency, low communication overhead.
- ✓ Can protect both data and model.
- ✓ Easier integration for existing models.

Cons:

- ✗ Requires trust in hardware vendor (Intel, ARM, etc.).
- ✗ Vulnerable to **side-channel attacks** (Spectre, Meltdown, etc.).
- ✗ Limited secure memory size (can be issue for large models).
- ✗ Requires specific hardware support.

Approach 4: Differential Privacy (DP)

Concept: Add carefully calibrated noise to computations to protect individual data points [12]. **Role in Inference:** Primarily protects **training data privacy**, not usually input privacy during inference.

- **DP Training:** Train a model (e.g., DP-SGD) that is inherently privacy-preserving w.r.t. its training data [13]. Queries to this model are safer.
- **Output Perturbation:** Server adds noise to the prediction $f(x, w)$ before sending it back. (Rarely desired due to accuracy loss).
- **Local DP:** Client adds noise to their input x *before* sending it. Protects input from server, but often causes large accuracy drop [14].

Pros:

- ✓ Mathematically rigorous privacy guarantees (ϵ-DP).
- ✓ Lightweight computation.
- ✓ Addresses training data leakage concerns.

Cons:

- ✗ Doesn't hide raw input from the server (unless local DP).
- ✗ Adding noise inherently reduces accuracy/utility.
- ✗ Primarily protects training set, not inference input itself.

Often used to complement crypto/TEE methods.

Approach 5: Split / Federated Inference

Concept: Don't send raw data, maybe send the model (or part of it) instead.

- **On-Device Inference:** Run the entire model locally on the client device.
  - \+ Perfect data privacy (data never leaves).
  - \- Model is fully exposed. Requires capable client device.
- **Split Inference / Split Learning:** [14], [15]
  1. Client runs first few layers of the model on raw input x.
  2. Client sends intermediate activation (feature vector) to Server.
  3. Server runs remaining layers, sends final prediction back.
  - \+ Hides raw data x. Balances computation. Lower overhead than crypto.
  - \- Intermediate activations can still leak information! Model is partially revealed.
  - \+ Can add noise (Local DP) to activations for better privacy (e.g., Split-and-Denoise [14]).

Comparison of Approaches I

Approach	Privacy Guarantee	Efficiency	Comm. Cost	Scalability	Security Assumptions	Limitations
Homomorphic Encryption (HE)	Strong (Mathematical)	Low (High Computation)	Low (Min Interaction)	Limited (Slow for Deep)	No Trusted Party	High Latency, Large Ciphertext
Secure Multi-Party Comp. (MPC)	Strong (Mathematical)	Medium (Protocol Dep.)	High (Multiple Rounds)	Moderate	Honest Majority / Non-collusion	High Comm. Overhead
Trusted Exec. Env. (TEEs)	Hardware-Based	High (Near-Native)	Low (Minimal Comm.)	High (Large Models OK)	Trusted Hardware Vendor	Side-Channels, Trust Vendor
Differential Privacy (DP)	Statistical (Training Data focus)	High	Low (Minimal Overhead)	High	Trusted Aggregator (if used)	Adds Noise → Utility Loss, Doesn't hide input
Federated / Split Learning	Partial (Limited Input Privacy)	High (On-Device part)	Medium (Intermediate Data)	High	Secure Aggregator (if FL)	Intermediate Leakage, Model Exposure

Table adapted from **baranski2024survey**.

Frameworks and Libraries

PPML is becoming more practical with dedicated tools:

Homomorphic Encryption:

- Microsoft SEAL [16]
- OpenFHE [17]
- HElib [18]
- TFHE / Concrete [19], [20]
- nGraph-HE (Intel) [21]

Secure Multi-Party Comp.:

- ABY / ABY3 [22]
- CrypTen (Meta/Facebook) [23]
- TF Encrypted [24]
- EMP-toolkit [25]
- SecureDFL [26] (Federated focus)

These tools abstract away some of the cryptographic complexity.

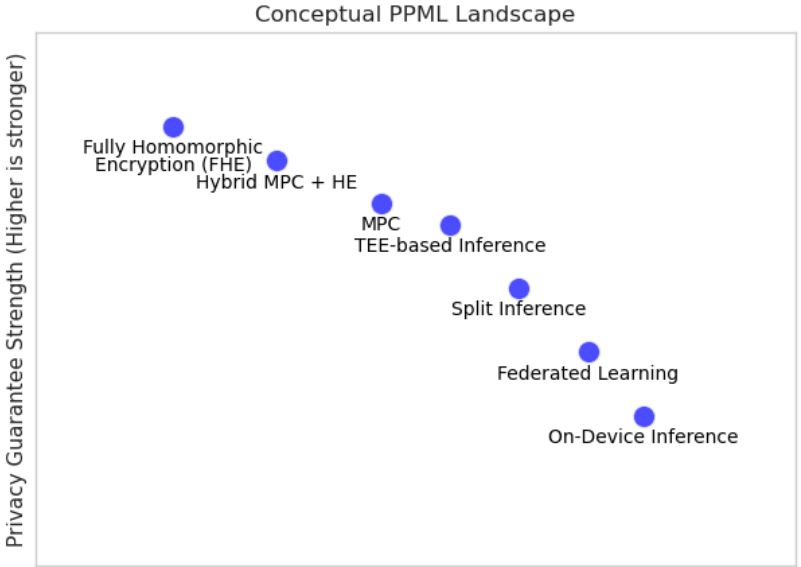
Performance: The Reality Check

- **Pure HE:** Still very slow for deep networks. CryptoNets (MNIST) took minutes initially [9], though optimized versions are faster. Logistic regression example: 5 sec/sample [27].
- **MPC / Hybrid:** Much better, but communication is the bottleneck.
  - Gazelle (small CNN): ¡ 0.5s latency, few MB comm [10].
  - Delphi (ResNet-32 optimized): Few seconds online time, tens of MB comm (after heavy offline pre-processing) [5].
  - Recent work (ResNet-18): 1 second latency with hardware optimization [11]. ReLU ops often dominate time [28].
- **TEEs:** Fastest option (near native speed), but relies on hardware trust.
- **Key Trend:** Optimizing the *model architecture* for privacy (fewer non-linear ops, quantization) significantly helps performance [5], [29].

Takeaway

There's a significant overhead compared to plaintext inference, but progress is rapid, especially with hybrid methods and optimization.

The PPML Landscape: Privacy vs. Efficiency



Big players solutions — Apple PCC

Problem: How to run powerful AI features (e.g., advanced Siri) that are too complex for on-device processing, without compromising user privacy? [30], [31]

Apple's Approach (PCC):

- **Secure Offloading:** Send necessary data to dedicated Apple servers for inference.
- **Not HE/MPC:** Computation happens on *decrypted* data inside the PCC infrastructure.
- **Focus on Infrastructure Security & Policy:**
  - **Stateless Servers:** Data is NOT stored persistently after processing. Used only for the query.
  - **Cryptographic Unlinkability:** Requests are not tied to a user's Apple ID on PCC servers.
  - **Minimal Footprint:** Purpose-built, minimal OS and software stack reduces attack surface.
  - **Access Control:** Technical measures claimed to prevent Apple employees from accessing user data during processing.

Key Idea

Privacy relies on strong infrastructure design and operational guarantees, not purely on cryptographic blindness during compute (like HE/MPC).

PCC Security: Verifiable Privacy Claims

How does Apple build trust in these infrastructure claims?

Verifiability Mechanism:

- **Published Software Images:** Apple makes the software images that run on PCC servers publicly available.
- **Independent Expert Audit:** Security researchers can download and analyze these images.
- **Goal of Audit:** Verify that the code implements the privacy promises (statelessness, no logging, unlinkability, etc.).
- **Contrast to Crypto Proofs:** This is *not* a per-computation proof (like ZKP). It's trust based on *transparency and expert validation* of the system's software design.

Trust Model

Users trust that:

1. The published software images accurately represent what runs on PCC servers.
2. The expert community's analysis is thorough and trustworthy.
3. Apple operates the infrastructure securely and honestly according to the verified design.

PCC in the PPML Landscape

Where does Apple's approach fit compared to the techniques we surveyed?

Efficiency:

✓ **High.** Compute on plaintext data on powerful servers. Much faster than HE/MPC. Comparable to standard cloud inference.

Privacy Guarantees:

- Stronger than typical cloud processing.
- Weaker mathematical guarantee than HE/MPC (requires trust in Apple's operation & security).
- Conceptually similar to TEEs (relies on hardware/software isolation).

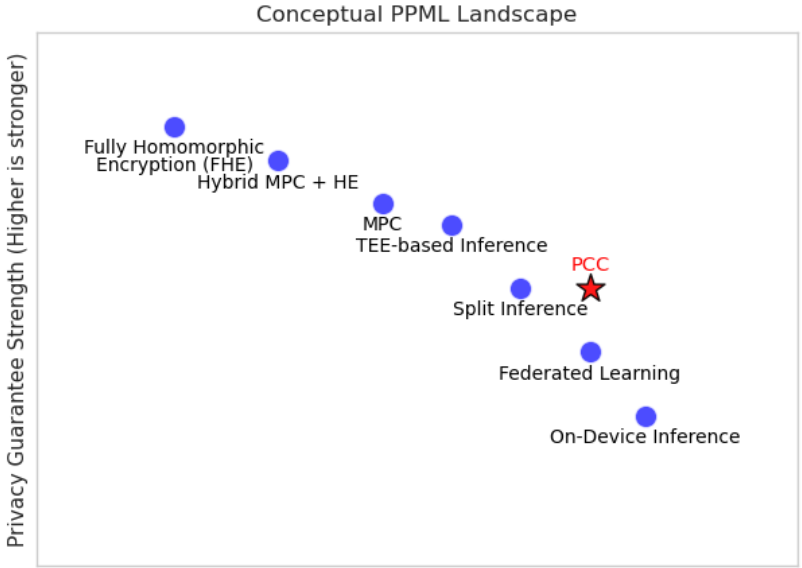
Conceptual Placement:

- Near TEEs on efficiency.
- Privacy level between standard cloud and TEE/Crypto, heavily reliant on:
  - Vendor operational security.
  - Verifiability via transparency.
- Represents a specific **"Vendor-Controlled Secure Infrastructure"** model.

Takeaway

Major players like Apple are developing custom, large-scale solutions that blend hardware security, operational policy, and transparency to balance strong privacy with high performance for demanding ML tasks.

The PPML Landscape with PCC



The PPML Landscape: Privacy vs. Efficiency

- No single "best" solution. Choice depends on the specific scenario.
- Spectrum from max efficiency/low privacy (on-device) to max privacy/high cost (FHE).
- Hybrids (MPC, TEE, Split) try to find a balance.

Open Challenges

- **Scalability:** Handling massive models (Transformers, LLMs) efficiently. Current methods struggle. Need model compression/distillation or new PPML techniques [32], [33].
- **Efficiency:** Reducing computation and communication (fewer rounds, hardware acceleration [7]).
- **Robustness:** Security against malicious actors (not just honest-but-curious). Defending against TEE side-channels.
- **Usability:** Making PPML tools "plug-and-play" for ML engineers. Need better compilers/frameworks [34], [35].
- **Privacy-Utility Trade-off:** Systematically balancing privacy guarantees vs. performance/accuracy needs.
- **Standardization:** Common benchmarks and evaluation methods needed for fair comparison [4].

Conclusion

- PPML inference is crucial for enabling secure MLaaS.
- We have a diverse toolkit: HE, MPC, TEEs, DP, Split Learning. Each with pros and cons.
- Hybrid approaches are currently state-of-the-art for balancing privacy and performance.
- Performance is improving, but significant overhead remains, especially for large models.
- Key challenges include scalability, usability, and robustness against stronger adversaries.
- Future directions: Better frameworks, hardware acceleration, standardized benchmarks, combining techniques (e.g., DP + MPC).

The Big Picture

PPML allows us to harness the power of AI on sensitive data, unlocking applications in healthcare, finance, and beyond, while respecting privacy. It's a rapidly evolving and important field!

Thank You!

Questions?

Based on the paper: Baranski, S. (2025). A Survey on Privacy-Preserving Machine Learning Inference. *TASK Quarterly*

References I

[1] *CSAM Detection - Technical Summary*, 2021. [Online]. Available: https://www.apple.com/child-safety/pdf/CSAM_Detection_Technical_Summary.pdf.

[2] A. Bhowmick and D. Boneh, *The Apple PSI System*, Jul. 29, 2021. [Online]. Available: https://www.apple.com/child-safety/pdf/Apple_PSI_System_Security_Protocol_and_Analysis.pdf.

[3] D. Evans, V. Kolesnikov, and M. Rosulek, "A pragmatic introduction to secure multi-party computation," *Foundations and Trends® in Privacy and Security*, vol. 2, no. 2–3, pp. 70–246, 2018, ISSN: 2474-1558.

[4] A. Guerra-Manzanares, L. J. L. Lopez, M. Maniatakos, and F. E. Shamout, "Privacy-preserving machine learning for healthcare: Open challenges and future perspectives," in vol. 13932, 2023, pp. 25–40. DOI: 10.1007/978-3-031-39539-0_3. arXiv: 2303.15563 [cs]. [Online]. Available: http://arxiv.org/abs/2303.15563 (visited on 03/05/2025).

[5] P. Mishra, R. Lehmkuhl, A. Srinivasan, W. Zheng, and R. A. Popa, "Delphi: A Cryptographic Inference Service for Neural Networks,", presented at the 29th USENIX Security Symposium (USENIX Security 20), 2020, pp. 2505–2522, ISBN: 978-1-939133-17-5. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/mishra (visited on 03/06/2025).

[6] C. Zhang and S. Li, "State-of-the-Art Approaches to Enhancing Privacy Preservation of Machine Learning Datasets: A Survey," arXiv: 2404.16847 [cs]. (Jan. 28, 2025), [Online]. Available: http://arxiv.org/abs/2404.16847 (visited on 03/06/2025), pre-published.

References II

[7] B. Reagen, W. Choi, Y. Ko, *et al.*, "Cheetah: Optimizing and Accelerating Homomorphic Encryption for Private Inference," arXiv: 2006.00505 [cs]. (Oct. 8, 2020), [Online]. Available: http://arxiv.org/abs/2006.00505 (visited on 03/06/2025), pre-published.

[8] R. Xu, N. Baracaldo, and J. Joshi, "Privacy-Preserving Machine Learning: Methods, Challenges and Directions," arXiv: 2108.04417 [cs]. (Sep. 22, 2021), [Online]. Available: http://arxiv.org/abs/2108.04417 (visited on 03/05/2025), pre-published.

[9] N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16, New York, NY, USA: JMLR.org, Jun. 19, 2016, pp. 201–210.

[10] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "Gazelle: A Low Latency Framework for Secure Neural Network Inference," arXiv: 1801.05507 [cs]. (Jan. 16, 2018), [Online]. Available: http://arxiv.org/abs/1801.05507 (visited on 03/06/2025), pre-published.

[11] J. Mo, K. Garimella, N. Neda, A. Ebel, and B. Reagen, "Towards Fast and Scalable Private Inference," in *Proceedings of the 20th ACM International Conference on Computing Frontiers*, Bologna Italy: ACM, May 9, 2023, pp. 322–328, ISBN: 9798400701405. DOI: 10.1145/3587135.3592169. [Online]. Available: https://dl.acm.org/doi/10.1145/3587135.3592169 (visited on 03/06/2025).

[12] N. Papernot, M. Abadi, Ú. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data," arXiv: 1610.05755 [stat]. (Mar. 3, 2017), [Online]. Available: http://arxiv.org/abs/1610.05755 (visited on 03/05/2025), pre-published.

References III

[13] X. Li, F. Tramèr, P. Liang, and T. Hashimoto, "Large Language Models Can Be Strong Differentially Private Learners," arXiv: 2110.05679 [cs]. (Nov. 10, 2022), [Online]. Available: http://arxiv.org/abs/2110.05679 (visited on 03/05/2025), pre-published.

[14] P. Mai, R. Yan, Z. Huang, Y. Yang, and Y. Pang, "Split-and-Denoise: Protect large language model inference with local differential privacy," arXiv: 2310.09130 [cs]. (Aug. 27, 2024), [Online]. Available: http://arxiv.org/abs/2310.09130 (visited on 03/06/2025), pre-published.

[15] X. Yang, J. Sun, Y. Yao, J. Xie, and C. Wang, "Differentially Private Label Protection in Split Learning," arXiv: 2203.02073 [cs]. (Mar. 4, 2022), [Online]. Available: http://arxiv.org/abs/2203.02073 (visited on 03/11/2025), pre-published.

[16] *Microsoft/SEAL*, Microsoft, Mar. 5, 2025. [Online]. Available: https://github.com/microsoft/SEAL (visited on 03/06/2025).

[17] A. Al Badawi, J. Bates, F. Bergamaschi, *et al.*, "Openfhe: Open-source fully homomorphic encryption library," in *Proceedings of the 10th Workshop on Encrypted Computing & Applied Homomorphic Cryptography*, ser. WAHC'22, Los Angeles, CA, USA: Association for Computing Machinery, 2022, pp. 53–63. DOI: 10.1145/3560827.3563379. [Online]. Available: https://doi.org/10.1145/3560827.3563379.

[18] *Homenc/HElib*, homenc, Mar. 11, 2025. [Online]. Available: https://github.com/homenc/HElib (visited on 03/12/2025).

[19] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, *TFHE: Fast fully homomorphic encryption library*, https://tfhe.github.io/tfhe/, August 2016.

[20] Zama, *Concrete: TFHE Compiler that converts python programs into FHE equivalent*, https://github.com/zama-ai/concrete, 2022.

References IV

[21] F. Boemer, Y. Lao, R. Cammarota, and C. Wierzynski, "nGraph-HE: A graph compiler for deep learning on homomorphically encrypted data," in *Proceedings of the 16th ACM International Conference on Computing Frontiers*, ser. CF '19, New York, NY, USA: Association for Computing Machinery, Apr. 30, 2019, pp. 3–13, ISBN: 978-1-4503-6685-4. DOI: 10.1145/3310273.3323047. [Online]. Available: https://doi.org/10.1145/3310273.3323047 (visited on 03/12/2025).

[22] P. Rindal, *The ABY3 Framework for Machine Learning and Database Operations*. https://github.com/ladnir/aby3.

[23] B. Knott, S. Venkataraman, A. Hannun, S. Sengupta, M. Ibrahim, and L. van der Maaten, "CrypTen: Secure Multi-Party Computation Meets Machine Learning," arXiv: 2109.00984 [cs]. (Sep. 15, 2022), [Online]. Available: http://arxiv.org/abs/2109.00984 (visited on 03/06/2025), pre-published.

[24] *Tf-encrypted/tf-encrypted*, TF Encrypted, Mar. 2, 2025. [Online]. Available: https://github.com/tf-encrypted/tf-encrypted (visited on 03/12/2025).

[25] X. Wang, A. J. Malozemoff, and J. Katz, *EMP-toolkit: Efficient MultiParty computation toolkit*, https://github.com/emp-toolkit, 2016.

[26] B. Jeon, S. M. Ferdous, M. R. Rahman, and A. Walid, "Privacy-preserving Decentralized Aggregation for Federated Learning," arXiv: 2012.07183 [cs]. (Dec. 28, 2020), [Online]. Available: http://arxiv.org/abs/2012.07183 (visited on 03/12/2025), pre-published.

References V

[27] "Enable fully homomorphic encryption with Amazon SageMaker endpoints for secure, real-time inferencing — AWS Machine Learning Blog," (Mar. 23, 2023), [Online]. Available: https://aws.amazon.com/blogs/machine-learning/enable-fully-homomorphic-encryption-with-amazon-sagemaker-endpoints-for-secure-real-time-inferencing/ (visited on 03/06/2025).

[28] L. Zhou, Z. Wang, H. Cui, Q. Song, and Y. Yu, "Bicoptor: Two-round Secure Three-party Non-linear Computation without Preprocessing for Privacy-preserving Machine Learning," arXiv: 2210.01988 [cs]. (Apr. 19, 2024), [Online]. Available: http://arxiv.org/abs/2210.01988 (visited on 03/06/2025), pre-published.

[29] P. Mohassel and Y. Zhang, "SecureML: A System for Scalable Privacy-Preserving Machine Learning," (2017), [Online]. Available: https://eprint.iacr.org/2017/396 (visited on 03/12/2025), pre-published.

[30] "Blog - Private Cloud Compute: A new frontier for AI privacy in the cloud - Apple Security Research," Blog - Private Cloud Compute: A new frontier for AI privacy in the cloud - Apple Security Research. (Oct. 6, 2024), [Online]. Available: https://security.apple.com/blog/private-cloud-compute/ (visited on 04/08/2025).

[31] Apple, director, *WWDC 2024 — June 10 — Apple*, Jun. 10, 2024. [Online]. Available: https://www.youtube.com/watch?v=RXeOiIDNNek (visited on 04/08/2025).

References VI

[32] A. Polino, R. Pascanu, and D. Alistarh, "Model compression via distillation and quantization," arXiv: 1802.05668 [cs]. (Feb. 15, 2018), [Online]. Available: http://arxiv.org/abs/1802.05668 (visited on 03/06/2025), pre-published.

[33] N. Dhyani, J. Mo, M. Cho, *et al.*, "PriViT: Vision Transformers for Fast Private Inference," arXiv: 2310.04604 [cs]. (Oct. 6, 2023), [Online]. Available: http://arxiv.org/abs/2310.04604 (visited on 03/06/2025), pre-published.

[34] N. Kumar, M. Rathee, N. Chandran, D. Gupta, A. Rastogi, and R. Sharma, "CrypTFlow: Secure TensorFlow Inference," (2019), [Online]. Available: https://eprint.iacr.org/2019/1049 (visited on 03/06/2025), pre-published.

[35] N. Chandran, D. Gupta, A. Rastogi, R. Sharma, and S. Tripathi, "Ezpc: Programmable and efficient secure two-party computation for machine learning," in *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, 2019, pp. 496–511.