FATE vs. SecretFlow: A Practical Comparison for Privacy-Preserving Machine Learning

Privacy-Preserving Data Analytics

Author: Vlad Ionita (V.Ionita@student.tudelft.nl)
Associate Professor: Dr. Zeki Erkin
Supervisor: Dr. Roland Kromes
Affiliation: TU Delft, Faculty of EEMCS

TU Delft

1. Background Information

Privacy-preserving machine learning (PPML) relies on secure techniques such as secure multi-party computation (SMPC) and federated learning (FL) to enable collaborative analytics without compromising data privacy.

SMPC allows multiple parties to jointly compute over private inputs without revealing them, using cryptographic primitives like secret sharing. For instance, Shamir's Secret Sharing [1] splits a secret into shares distributed among parties, and only a threshold number of shares is needed to reconstruct the secret, ensuring that partial information reveals nothing.

In contrast, FL keeps raw data local. Instead of sharing data, clients train models on their private data and send updates to a central server, which aggregates them to build a global model. FL can be horizontal (same features, different users) or vertical (same users, different features), and often incorporates SMPC or similar methods to enhance privacy.

Building on these techniques, this study compared two open-source frameworks: SecretFlow [2] and FATE [3]. Both support secure computation and federated analytics, but differ in architecture, supported protocols, deployment complexity, and target use cases.

2. Research Questions

The main question to answer is:

How do privacy-preserving machine learning frameworks such as SecretFlow and FATE implement secure computation techniques, and how do they compare in terms of ease of integration and scalability for collaborative data analysis tasks?

With the following subquestions:

1. **How do these tools make use of privacy-preserving approaches?**
2. **What is the difference between these tools in terms of ease of integration and scalability?**

3. Methodology

The chosen frameworks were selected for their contrasting design goals: SecretFlow prioritizes modularity and flexibility for research use, while FATE targets production environments with coordinated workflows and regulatory compliance. Their architectures and secure computation mechanisms were compared based on official documentation and intended use cases.

To assess their usability and performance, both frameworks were installed using the SURF Research Cloud [4] and tested using a federated logistic regression task on the Diagnostic Wisconsin Breast Cancer dataset [5]. Evaluation metrics consisted of setup complexity, documentation quality, implementation effort, and availability of secure computation modules.

5. Limitations & Future Work

Limited resources prevented full deployment of components like SecretFlow's SPU, so some evaluations relied on documentation and benchmarks. The study focused on SMPC, with less attention to other techniques like homomorphic encryption (HE) and differential privacy (DP). A small, homogeneous dataset also limited insights into diverse, large-scale scenarios. Future work should involve multi-node deployments with large and diverse datasets, as well as compare full architectural stacks, including HE and DP.

6. References

[1] Adi Shamir. How to Share a Secret. *Commun. ACM*, 22(11):612–613, 1979.
[2] WeBank AI Department. FATE: Federated AI Technology Enabler. https://fate.fedai.org, 2023. Accessed: May 2025.
[3] Ant Group. SecretFlow: A Unified Privacy-Preserving Computing Framework. https://www.secretflow.org.cn/en/, 2025. Accessed: May 2025.
[4] SURF. SURF Research Cloud. https://www.surf.nl/en/surf-research-cloud, 2024. Accessed: May 2025.
[5] William H. Wolberg, Olvi L. Mangasarian, and W. Nick Street. Breast Cancer Wisconsin (Diagnostic). https://doi.org/10.24432/C5DW2B, October 1995.
[6] Ant Group. SecretFlow Benchmark Results. https://www.secretflow.org.cn/en/docs/secretflow/v1.12.0b0/developer/benchmark/overall_benchmark, 2024. Accessed: June 2025.
[7] Ant Group. Performance Analysis: SecureBoost vs XGBoost. https://www.secretflow.org.cn/en/docs/secretflow/v1.12.0b0/developer/benchmark/sgb_benchmark, 2024. Accessed: June 2025.
[8] Zelei Liu, Yuanyuan Chen, Yansong Zhao, Han Yu, Yang Liu, et al. Contribution-Aware Federated Learning for Smart Healthcare. In *AAAI*, pages 12396–12404. AAAI Press, 2022.
[9] Aristeidis Karras, Anastasios Giannaros, Leonidas Theodorakopoulos, et al. FLIBD: A Federated Learning-Based IoT Big Data Management Approach for Privacy-Preserving over Apache Spark with FATE. *Electronics*, 12(22):4633, 2023.

4. Results

Protocol	Security Mode	Num. Parties	Computation Complexity	Communication Overhead	Use Case	Drawbacks
ABY3	Semi Honest	3	Moderate: bit-level operations, fixed-point arithmetic	Moderate bandwidth due to replicated secret sharing	Training and inference in ML, efficient fixed-point arithmetic	Limited to 3 parties, fixed trust assumption
Semi2k SPDZ	Semi Honest	2+	Moderate - High: expensive MAC generation, requires offline preprocessing	High bandwidth due to pre-shared randomness and authentication tags	General-purpose MPC, supports active security in SPDZ variant	Communication-heavy, requires pre-processing and MAC generation
Cheetah	Semi Honest	2	Low: SIMD-optimized, minimal interactive operations	Very low, batch-friendly vectorized design	Efficient 2-party ML inference	Optimized for 2-party use only, not as flexible for n-party setups
SPDZ	Malicious	2+	High: expensive FHE-based preprocessing, complex arithmetic	Very high, significant bandwidth for MACs and ciphertexts	General-purpose secure computation under strong adversaries	Preprocessing-heavy, slower runtime, high bandwidth needs
Feldman VSS	Semi Honest	n	Low: polynomial interpolation/evaluation	Low, public verifiability adds small overhead	Secret sharing schemes, threshold cryptography	Public verifiability but no hiding of shares from dealer
ECDH PSI	Semi Honest	2	Low: scalar EC operations, linear with input size	Low due to lightweight cryptographic primitives	Two-party PSI such as user ID matching	Relies on elliptic curve assumptions, not optimized for large-scale datasets
KKRT PSI	Semi Honest	2	Low: OT extension and hashing, efficient for large inputs	Low - moderate depending on input set size	High-speed PSI on medium to to large datasets	Performance may degrade with very large input sets
BC22PCG PSI	Malicious	2	Low - Moderate: pseudorandom code-based, higher per-element cost	Moderate, includes additional checks for malicious behavior	High-security PSI with better fault tolerance	More complex implementation, slightly higher overhead than OT-based PSI

Figure 1: Secure computation protocols and components used by each framework

Protocols and components used by SecretFlow:

- SMPC protocols: ABY3, Semi2k-SPDZ, Cheetah
- PSI protocols: ECDH-PSI, KKRT-PSI, BC22PCG-PSI

Protocols and components used by FATE:

- SMPC protocol: SPDZ
- PSI protocol: ECDH-PSI
- Cryptographic primitive: Feldman VSS

Integration Evaluation

SecretFlow provided a smoother integration experience due to its well-maintained documentation, clear tutorials, and developer-friendly tools, making it ideal for academic and experimental use. In contrast, FATE's documentation was fragmented and occasionally outdated, resulting in a more challenging learning curve during setup.

Scalability Evaluation

SecretFlow's SPU backend is optimized for extensibility and parallel computation. Benchmarks [6, 7] suggest that it scales well with increasing data volume and parties. FATE supports production-scale deployments with built-in coordination, monitoring tools, and native handling of vertically partitioned data, proven in real-world applications like healthcare [8] and smart cities [9].