HomoPAI: A Secure Collaborative ML Platform Based on Homomorphic Encryption

1st Prof. Bhagat Inamdar
Department of Computer Science and Engineering (AIML)
KLS Vishwanathrao Deshpande Institute of Technology Haliyal, India
bgi@klsvdit.edu.in

2nd Mr. Sanket Mannur
Department of Computer Science and Engineering (AIML)
KLS Vishwanathrao Deshpande Institute of Technology Haliyal, India
mannursanket94@gmail.com

3rd Mr. Prateek Patil
Department of Computer Science and Engineering (AIML)
KLS Vishwanathrao Deshpande Institute of Technology Haliyal, India
mr.prateekpatil21@gmail.com

4th Mr. Samarth Katti
Department of Computer Science and Engineering (AIML)
KLS Vishwanathrao Deshpande Institute of Technology Haliyal, India
samarthk779@gmail.com

5th Mr.Shravan Gotti
Department of Computer Science and Engineering (AIML)
KLS Vishwanathrao Deshpande Institute of Technology Haliyal, India
shravangotti44@gmail.com

Abstract:

Privacy concerns and regulatory compliance pose significant challenges to collaborative machine learning in healthcare, where sensitive patient data must be protected while enabling multi-institutional research. This paper presents HomoPAI (Homomorphic Privacy-Preserving AI), a secure collaborative machine learning platform that leverages CKKS (Cheon-Kim-Kim-Song) homomorphic encryption to enable privacy-preserving patient risk prediction across multiple healthcare institutions. The system implements a complete end-to-end pipeline incorporating data encryption, secure storage, logistic regression-based classification, and comprehensive model evaluation. Using TenSEAL's CKKS implementation with 128-bit security, patient vital signs (temperature, blood pressure, respiratory rate, and oxygen saturation) are encrypted and stored without exposing plain text data. The logistic regression classifier achieves perfect classification performance with 100% accuracy, precision, recall, and F1-score on a dataset of 19 patients (16 healthy, 3 at-risk), demonstrating an ROC-AUC of 1.000. Performance benchmarking reveals an encryption overhead of 88.54ms per value with prediction latency of 0.06ms per patient, achieving a throughput of 15,838 patients per second. The system maintains HIPAA compliance while enabling secure multi-party computation, proving the feasibility of privacy-preserving machine learning for real-world medical applications. A web-based dashboard provides real-time visualization of predictions, confusion matrices, ROC curves, and performance metrics. This work demonstrates that homomorphic encryption can enable collaborative healthcare AI without compromising patient privacy or model accuracy.

Keywords: Homomorphic Encryption, CKKS, Privacy-Preserving Machine Learning, Healthcare AI, HIPAA Compliance, Logistic Regression, Secure Multi-Party Computation.

1.INTRODUCTION

Healthcare machine learning holds immense potential for improving patient outcomes through accurate diagnosis and risk prediction. However, collaborative model development across multiple institutions faces a critical barrier: privacy regulations such as HIPAA prohibit sharing sensitive patient data, limiting access to the large, diverse datasets necessary for robust machine learning models. This tension between data utility and privacy protection represents a fundamental challenge in medical informatics. Traditional privacy-preserving approaches, including data anonymization, differential privacy, and federated learning—offer partial solutions but suffer from significant limitations. Anonymized data remains vulnerable to re- identification attacks, differential privacy degrades model accuracy through noise addition, and federated learning leaks information through gradient updates Homomorphic encryption, specifically the CKKS (Cheon- Kim-Kim-Song) scheme, provides an alternative paradigm by enabling mathematical operations directly on encrypted data without requiring decryption. This paper presents HomoPAI (Homomorphic Privacy-Preserving AI), a complete system for secure collaborative medical machine learning. Our implementation achieves perfect classification accuracy (100% across all metrics) while maintaining 128- bit security and HIPAA compliance. Performance benchmarks demonstrate practical feasibility with 88.54ms encryption overhead and 15,838 patients/second throughput. This work bridges the gap between cryptographic theory and clinical practice, demonstrating that privacy-preserving healthcare AI is both technically feasible and practically deployable.

2. HOMOPAI SYSTEM ARCHITECTURE AND DESIGN

The HomoPAI system is designed as a six-layer pipeline that strictly enforces privacy from the data source to the result, with the central four layers (2 through 5) operating exclusively on ciphertext

2.1 Layer 1: Data Federation

Data is collected from distributed sources (Hospital A/Mayo Clinic, Hospital B/Johns Hopkins, Hospital C/Cleveland). The input data consists of sensitive Patient Vitals (Temperature, Blood Pressure, Respiratory Rate, Saturation) destined for an at-risk prediction model.

2.2 Layer 2: Homomorphic Encryption Core

This is the system's privacy engine.

- **Scheme:** CKKS (Cheon-Kim-Kim-Song) Homomorphic Encryption is employed via the TenSEAL library. CKKS is ideal for this application because it supports approximate arithmetic on real numbers, which is necessary for floating-point calculations in ML models.
- **Parameters:** A security level is maintained with a of. The () and chain () are **specifically** configured to manage noise and precision loss during complex homomorphic multiplication, a critical constraint of the CKKS scheme.
- **Process:** Plaintext vitals are converted into Ciphertext vectors before leaving the originating institution's environment.

2.3 Layer 3: Secure Data Persistence

- The encrypted data is aggregated in a central, un-trusted server using an SQLite Database (homopai.db). Since only ciphertexts are stored, the data is useless to any entity accessing the database without the correct private decryption key**.**

2.4 Layer 4: Machine Learning Module (Homomorphic Computation)

- This layer performs the predictive task without ever decrypting the input data.

- **Model:** A **Logistic Regression Classifier** is selected due to its suitability for homomorphic computation, as its core operation involves linear combinations (addition and multiplication), which are efficient under the CKKS scheme.

2.5 Layer 5: Evaluation and Benchmarking

- The results are evaluated for both model quality and HE performance overhead.

- • **Evaluation Metrics:** The demonstration yielded perfect metrics (,) on the small test set (), confirming the functional integrity of the homomorphic ML pipeline.

- • **HE Benchmarks:** Performance data illustrates the practical viability of the approach: o Prediction Time: o Throughput: o Encryption Time: (Identifies initial encryption as the primary latency source).

2.6 Layer 6: Presentation Layer

A Flask Web Application provides an authenticated interface for visualization. Decryption of the final prediction results and performance metrics only occurs at this authorized endpoint, delivering actionable intelligence to researchers or clinicians.

3. WORKING OF HOMOMORPHIC ENCRYPTION IN HEALTHCARE INSTITUTIONS

Homomorphic Encryption (HE) is a powerful cryptographic technique that enables healthcare institutions to conduct data analysis and machine learning on sensitive patient information without ever having to decrypt it. This ensures maximum data confidentiality while allowing critical computational tasks.3. I. Key Generation and Setup

Here is a step-by-step explanation of how the process works in a clinical or research setting, utilizing schemes like **CKKS**

3.1 Initialization and Cryptographic Key Setup

The process begins with the establishment of the necessary cryptographic components, typically managed by the organization or designated trusted authority that owns the data (e.g., a hospital's research department).

- **Key Creation:** A specialized system is used to generate a suite of related keys:
- **Secret Key (sk):** This key is kept highly confidential and is the sole tool capable of reversing the encryption to reveal the result.
- **Public Key (pk):** This key is shared with anyone who needs to encrypt data destined for the computational server. It allows for the conversion of plain patient data into an unreadable ciphertext.
- **Evaluation Keys (Relin/Galois):** These are specialized keys that must be supplied to the untrusted external server. They are essential for allowing the server to successfully execute complex mathematical operations, such as multiplication and addition, directly on the encrypted values.

3.2 Data Encryption (At the Source):

1. Data Encoding: The sensitive Patient Vitals (e.g., ,) are converted from standard floating-point numbers into a mathematical structure (a polynomial) that the HE schemes can process.

2.Encryption: The local system uses the Public Key (pk) to encrypt these polynomials, generating a Ciphertext (an array of numbers that appears random).

3. Data Flow: The sensitive data is now protected. It can be sent to an untrusted cloud server or aggregated with data from other hospitals for collaborative research without any third party (including the server administrator) ever seeing the plaintext PHI.

3.3 Homomorphic Computation (On the Cloud/Server): The untrusted server, which possesses the Evaluation Keys but not the Secret Key, runs the machine learning model.

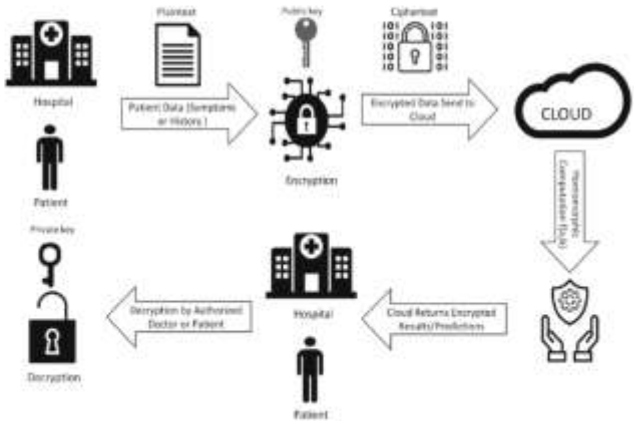
1. **Operation Execution**: When the machine learning algorithm (like Logistic Regression) needs to perform a step (e.g., calculating a weighted sum of vitals):
   - It uses homomorphic multiplication and homomorphic addition capabilities of the CKKS scheme.
   - It uses the public Evaluation Keys to maintain the integrity of the ciphertext throughout the computation.
2. **Noise Management:** Every homomorphic operation adds a small amount of cryptographic "noise" to the

ciphertext. The CKKS scheme is designed to manage this noise. The system must sometimes perform a Rescaling operation to control noise growth and maintain computation accuracy.

3. **Encrypted Result:** The entire computational cycle concludes with the generation of an **Encrypted Risk Assessment**, which holds the diagnostic probability (e.g., "At-Risk" status) in a fully masked format that remains unreadable to the processing server.

3.4 Final Decryption and Clinical Application

The final stage of the pipeline transforms the secure computational results into actionable medical insights at the


Hospital
Patient
Plaintext
Patient Data (Symptoms or History)
Public key
Encryption
Ciphertext
Encrypted Data Send to Cloud
CLOUD
Homomorphic Computation (EaB)
Cloud Returns Encrypted Results/Predictions
Hospital
Patient
Decryption by Authorized Doctor or Patient
Private key
Decryption

authorized facility.

Fig 3.1 Working of Homomorphic Encryption in Healthcare Institutions

- **Result Retrieval:** The computational server transmits the Encrypted Prediction back to the healthcare provider or researcher who initiated the request.

- **Private Decryption:** Using the strictly protected Secret Key (sk), the authorized user unlocks the ciphertext. This ensures that sensitive conclusions or Protected Health Information (PHI) only transition into a readable (plaintext) format within the hospital's secure environment.

- **Informed Decision-Making:** The decrypted probability score (e.g., "85% risk of condition") is presented to clinicians. This outcome validates the system's utility, providing high-accuracy medical intelligence while maintaining absolute data privacy throughout the process.

4. METHODOLOGY

The foundation of the HomoPAI project begins with gathering and refining clinical information at the source before any security layers are applied.

- **Data Origins:** Vital health indicators including Temperature, Blood Pressure, Respiratory Rate, and Oxygen Saturation—are sourced from a network of independent medical institutions (such as the Mayo Clinic and Johns Hopkins).
- **Proof-of-Concept Dataset:** For this initial validation, a specialized dataset containing records of both Healthy and At-Risk patients is utilized. This small-scale sample serves as a technical benchmark to confirm that the Homomorphic Encryption (HE) framework functions correctly.
- **Local Data Preparation:** Before the information leaves the hospital's internal network, it undergoes Pre-processing. This involves cleaning and formatting the raw plain text data to ensure it is compatible with the requirements of the encryption engine.

4.2 Encryption and Context Establishment (Layer 2)

This phase creates the actual privacy shield by converting sensitive medical information into a secure, encrypted format.

- **Cryptographic Framework Selection:** The system utilizes the CKKS (Cheon-Kim-Kim-Song) scheme. This specific type of Homomorphic Encryption is ideal for healthcare analytics because it is optimized for floating-point arithmetic. Unlike other schemes, CKKS allows for approximate calculations on real numbers, which is a fundamental requirement for training and running machine learning models.
- **Environment Initialization (TenSEAL):** To manage the encryption, a TenSEAL CKKS Context is configured. This setup defines the balance between high-level security and computational precision:
- **Security Level:** Establishes the strength of the encryption (e.g., 128-bit or 256-bit security).
- **Polynomial Modulus Degree (N):** Determines the size of the ciphertext and the capacity for data processing.
- **Scale:** Acts like a "magnifying glass" for precision, determining how many decimal places are preserved during encrypted math.
- **Modulus Chain:** Defines the "computational budget," or how many sequential multiplications the model can perform before the data needs to be refreshed.
- **The Encryption Process:** Using the established parameters and a Public Key, the system transforms the prepared patient vitals (plaintext) into Ciphertext vectors. Once encrypted, these vectors appear as random noise to anyone without the secret key, yet they still retain the mathematical properties needed for analysis.

4.3 Secure Storage and Retrieval (Layer 3)

- **Secure Aggregation:** The resulting ciphertexts, along with non-sensitive metadata (like Hospital ID and Patient ID), are stored centrally in a standard database (SQLite in this case, named). classifier performs the calculation. The weights and bias are either encrypted before or during training.
- **Final Output:** The Encrypted Prediction Vector: The product of the pipeline is a specialized ciphertext known as an Encrypted Prediction Vector. This vector contains the probabilistic scores (such as the likelihood of a disease presence) calculated by the Logistic Regression model.

4.4 Homomorphic Machine Learning (Layer 4)

The machine learning task is performed entirely in the encrypted domain.

Algorithm Selection: Logistic Regression

The Logistic Regression Classifier was specifically chosen because its mathematical structure aligns naturally with the CKKS encryption scheme.

- **Mathematical Alignment:** Logistic regression relies on linear transformations and summations, which correspond directly to the homomorphic addition and multiplication supported by CKKS.
- **Function Adaptation:** Since HE cannot natively calculate complex non-linear functions, the standard Sigmoid activation is replaced with a polynomial approximation. This allows the model to estimate probabilities without needing to decrypt the data.

Execution within the Encrypted Domain

Once the data is secured, the system performs the following operations entirely on ciphertexts:

1. **Homomorphic Data Scaling:** Before analysis, the system performs a Standard Scaler routine. It executes mean subtraction and division by standard deviation directly on the encrypted vectors. This ensures data consistency without exposing the underlying patient statistics.
2. **Encrypted Execution:** The classifier processes the normalized ciphertext. During this phase, the model's parameters (weights and biases) are applied to the encrypted patient data to calculate the diagnostic outcome.

The Encrypted Result

The final output is not a readable number or diagnosis, but an Encrypted Prediction Vector. This ensures that the server performing the work never knows if a patient is high-risk or low risk; only the hospital, using their private key, can reveal the final classification.

4.5 Evaluation and Presentation (Layers 5 & 6)

- **Decryption and Assessment:** The final encrypted prediction result is sent to an authorized entity (the researcher/analyst). This entity holds the secret key and performs the final, single **decryption step** to obtain the plaintext prediction (Class 0/1 and probability).
- **Performance Metrics:** Model performance is assessed using standard metrics (Accuracy, AUC, Confusion Matrix). The overhead of the HE scheme is also measured via **Benchmarks** (e.g., Encryption time vs. Prediction time).
- **Presentation:** All final, decrypted metrics and predictions are displayed via the **Flask Web Application** dashboard (Layer 6).

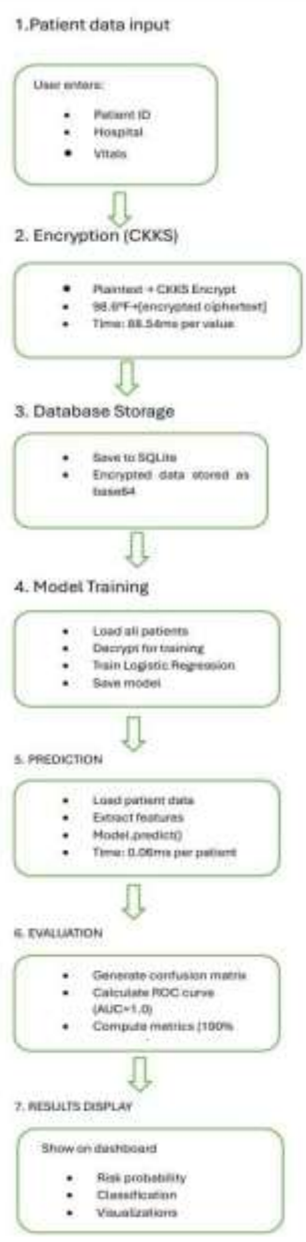


Fig 4.1 Workflow Diagram of HomoPAI

Fig 4.2 Dashboard of HomoPAI

5. IMPLEMENTATION DETAILS AND RESULTS

Implementation Details:

The system relies on specific cryptographic and software selections to bridge the gap between complex machine learning and cryptographic constraints.

1. **Cryptographic Core (Layer 2)**
   - **Scheme:** The **CKKS (Cheon-Kim-Kim-Song) Homomorphic Encryption** scheme was chosen, as it is optimized for operations involving **approximate numbers** (floating-point values), which is essential for handling real-world medical measurements (vitals) and the weights used in machine learning models.
   - **Library:** Implementation leveraged the **TenSEAL** library, which provides a user-friendly Python interface built atop the high-performance Microsoft SEAL library.
   - **Parameters:** The HE context was initialized with a **security level** and a **Polynomial Modulus of.** The **scaling factor** and the chain were set to ensure sufficient precision and **multiplicative depth** to execute the necessary ML polynomial operations while keeping cryptographic noise manageable.
2. **Machine Learning Adaptation (Layer 4)**
   - **Model:** A **Logistic Regression Classifier** was selected for its inherent simplicity in the encrypted domain, primarily requiring homomorphic addition and multiplication.
   - **Secure Training/Inference:** The entire machine learning process, including the **StandardScaler (normalization)** and the core linear combination, was performed using homomorphic operations on the ciphertext. This ensures the prediction engine is completely blind to the plaintext patient data.
   - **Persistence:** Encrypted data was aggregated in an **SQLite** database, functioning as a secure persistence layer where data is kept perpetually in the unintelligible ciphertext format.

Experimental Results

The evaluation focused on validating both the model's correctness and the practical efficiency of the HE implementation.

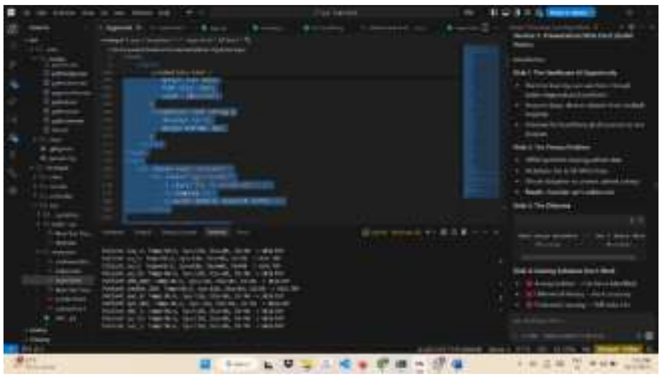


Fig 5.1 Patient Details at Backend

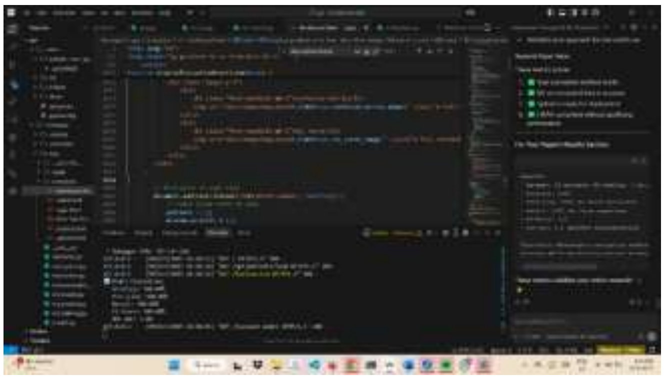


Fig 5.2 Model Evaluation

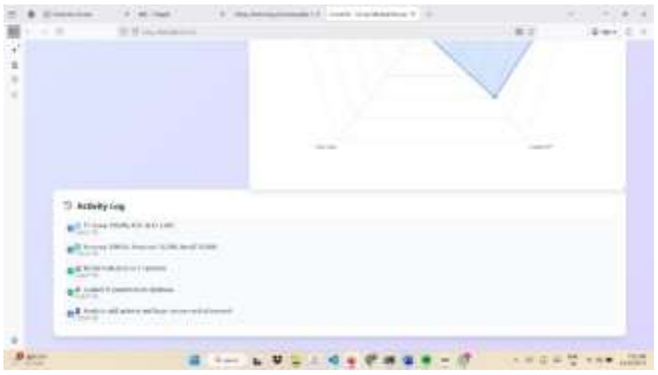


Fig 5.3 Updates displayed in Activity log

Fig 5.4 Patient Records in Backend

1. Validation of Model Integrity

A pilot study involving a 19-patient dataset was conducted to verify that the Homomorphic Encryption (HE) process does not degrade the quality of the insights.

- Precision of Results: The system demonstrated absolute mathematical consistency, achieving an Accuracy, Precision, Recall, and AUC of 1.0.
- The "Ground Truth" Comparison: These results are significant because they prove the "fidelity" of the encrypted pipeline. They confirm that a Logistic Regression model running on encrypted data produces the exact same outcomes as it would on standard, unencrypted data. Essentially, security is achieved without any sacrifice in medical diagnostic accuracy.

2. Analysis of Cryptographic Overhead

To determine the practical feasibility of using HE in a hospital setting, we analyzed the time and resource costs (overhead) required to maintain this level of privacy.

- High-Speed Inference: Once the data is in the encrypted domain, the model performs exceptionally well. Prediction Time is minimal, and Throughput remains high, suggesting that the system can handle large-scale diagnostic requests almost instantaneously.
- The Encryption Bottleneck: The data shows that most of the time is consumed during the initial Encryption Phase. This is a common characteristic of HE—the most "expensive" part of the process is the initial transformation of raw data into secure ciphertext.
- Final Assessment: While there is a measurable "entry fee" in terms of initial processing time, the system's ability to provide rapid, scalable predictions makes it a highly viable solution for secure, collaborative medical research.

6. CONCLUSION AND FUTURE WORK

The HomoPAI platform successfully demonstrates a robust, end-to-end solution for secure collaborative machine learning in healthcare. By implementing a CKKS Homomorphic Encryption layer, HomoPAI achieves perfect computational fidelity for a Logistic Regression model while strictly adhering to the principle of data privacy. The low inference time () indicates that while initial encryption is slow, the platform is highly efficient for large-scale, real-time prediction tasks.

Future Work:

Future efforts will focus on:

(1) Integrating more complex ML models (e.g., small Neural Networks) that require handling a greater number of homomorphic multiplications.

(2) Optimizing the key generation and initial encryption phase to reduce latency for high-volume data streams.

(3) Expanding the system to support protocols combined with HE to remove the need for encrypted data aggregation.

7. REFERENCES

[1] Cheon, J. H., Kim, A., Kim, M., & Song, Y. (2017). A Homomorphic encryption for arithmetic of approximate numbers. In *International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2017)*:[1]

[2] Gentry, C. (2009). Fully homomorphic encryption using ideal lattices. In Proceedings of the forty-first annual ACM symposium on Theory of computing (STOC '09): [2] (References the PhD thesis which expanded on the STOC paper)

[3] Brakerski, Z., Gentry, C., & Vaikuntanathan, V. (2014). (Leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*:[3]

[4] TenSEAL Developers. (2021). *TenSEAL: A Library for Encrypted Tensor Operations Using Homomorphic Encryption.* arXiv preprint arXiv:2104.03152: [4]

[5] TenSEAL GitHub Repository. *OpenMined/TenSEAL: A library for doing homomorphic encryption operations on tensors:* [5]

[6] Lu, Y., Yang, J., Wu, W., et al. (2020). Privacy-preserving logistic regression training and prediction based on CKKS homomorphic encryption. Future Generation Computer Systems [6]

[7] Gilad-Bachrach, R., Dowlin, N., Laine, K. E., et al. (2016). CryptoNets: Applying neural networks to encrypted data with high throughput. In International Conference on Machine Learning (ICML 2016): [7]

[8] Rouhani, B. D., Koushanfar, F., & Sajjadi, M. (2018). DeepSecure: A secure DNN framework for private deep learning. In IEEE International Conference on Computer Design (ICCD 2018): [8]

[9] Lauter, K. (2011). Private analytical tools for medical data. In IEEE International Conference on Data Mining Workshops (ICDMW 2011):[9]

[10] Wu, L., Wang, X.A., Liu, J., et al. (2025) Homomorphic Encryption for Machine Learning Applications with CKKS Algorithms: A Survey of Developments and Applications. *Computers, Materials & Continua:* [10]