applied sciences

MDPI

Article

Analyzing Machine Learning Models for Activity Recognition Using Homomorphically Encrypted Real-World Smart Home Datasets: A Case Study

Hasina Attaullah *, Sanaullah Sanaullah and Thorsten Jungeblut

Department of Engineering and Mathematics, Bielefeld University of Sciences, 33619 Bielefeld, Germany; sanaullah@hsbi.de (S.S.); thorsten.jungeblut@hsbi.de (T.J.)
* Correspondence: hasina.attaullah@hsbi.de

Abstract: The era of digitization and IoT devices is marked by the constant storage of massive amounts of data. The growing adoption of smart home environments, which use sensors and devices to monitor and control various aspects of daily life, underscores the need for effective privacy and security measures. HE is a technology that enables computations on encrypted data, preserving confidentiality. As a result, researchers have developed methodologies to protect user information, and HE is one of the technologies that make it possible to perform computations directly on encrypted data and produce results using this encrypted information. Thus, this research study compares the performance of three ML models, XGBoost, Random Forest, and Decision Classifier, on a real-world smart home dataset using both with and without FHE. Practical results demonstrate that the Decision Classifier showed remarkable results, maintaining high accuracy with FHE and even surpassing its plaintext performance, suggesting that encryption can enhance model accuracy under certain conditions. Additionally, Random Forest showed efficiency in terms of execution time and low prediction errors with FHE, making it a strong candidate for encrypted data processing in smart homes. These findings highlight the potential of FHE to set new privacy standards, advancing secure and privacy-preserving technologies in smart environments.

Keywords: fully homomorphic encryption; machine learning; smart homes; IoT; neuronal models; privacy-preserving

Citation: Attaullah, H.; Sanaullah, S.; Jungeblut, T. Analyzing Machine Learning Models for Activity Recognition Using Homomorphically Encrypted Real-World Smart Home Datasets: A Case Study. *Appl. Sci.* **2024**, *14*, 9047. https://doi.org/10.3390/app14199047

Academic Editor: Hiroaki Kikuchi

Received: 26 August 2024
Revised: 23 September 2024
Accepted: 2 October 2024
Published: 7 October 2024

1. Introduction

The advancement of IoT technology has significantly transformed modern life, which makes it an essential part of daily routines across various sectors. Technologies such as edge device-based artificial intelligence (AI) are increasingly extensive, especially in personal gadgets and smart home environments. This generation of smart homes, equipped with devices like smart meters, fitness trackers, and controllers for interconnected houses and smart cities, is reshaping how we interact with our surroundings and advancing modern society [1]. In a smart home environment, conventional home equipment can be remotely connected and operated through these technologies. These devices and sensors collect extensive amounts of personal data, which, when integrated into various platforms, use analytics to analyze human behavior, simplify decision-making, and offer personalized recommendations [2]. For example, it includes motion-activated lighting, thermostats that adjust room temperature based on real-time occupancy, activity recognition systems for elderly care, and intelligent security systems that identify family members, promoting ease, security, and efficiency within the household. However, while these advancements provide significant benefits, the sensitive nature of the data being collected poses serious privacy and security concerns. This study aims to address these concerns by comparing the performance of machine learning models with and without homomorphic encryption in a smart home environment. By doing so, we aim to explore the trade-offs between security

and computational efficiency, filling the gap in existing research that has not sufficiently focused on secure yet efficient data handling in smart homes.

Similarly, as AI replaces humans, IoT and smart technologies are taking advantage of the smart generation [3]. Instead of being constrained to basic data transmitters, devices are now intelligent robots that can understand data, make intelligent decisions, and adjust their performance according to user conditions [4,5]. These advancements in technology and innovations provide significant benefits but also introduce new challenges, primarily regarding the *privacy and security* of the generated data. Each device in a smart environment acts as a data source, continuously generating and collecting information about the resident's identification, habits, and preferences. These data, while valuable for decisive analysis, can reveal personal details about an individual's daily life routines and even health status [2,6,7]. The fear of being findable is still real, and it is dual. Unauthorized access to sensitive data generated in smart environments and excessive use of collected data for analysis compromise users' privacy and comfort levels. The breach of privacy in the first case was due to insecure network devices used in smart environments and a breach of privacy by third parties while using them for analysis. As a result, there should be a mechanism to secure smart environment datasets against unauthorized access and ensure their confidentiality while being shared for data mining.

To this end, considering previously discussed privacy and security breaches, the scope of this research study is to investigate the implementation of HE to improve privacy in smart homes, with a focus on residents' data being used for analysis while remaining secure and private, as illustrated in Figure 1. The analysis includes anomaly detection, activity recognition, and voice assistance. In addition, the presented use case study results demonstrate the privacy of sensor-based datasets from the smart home using different ML models. From the practical implementation point of view, we used two well-known libraries, TFHE [8] and concrete-ml based on HE, to calculate the accuracy and execution time for different ML models. Using FHE with ML in smart homes, this research aims to strike a balance between the growing demand for smart home datasets and the critical need for the preservation of privacy. Moreover, the objective of the case study is as follows:

- Conduct a comprehensive analysis of privacy risks by investigating the gaps in handling real-world smart home datasets.
- Undertake a thoroughgoing study of Homomorphic Encryption (HE). Examine its principles, applications, and effects on secure data processing.
- Explore the integration of XGBoost, Decision Classifier, and Random Forest ML models with Fully Homomorphic Encryption (FHE) for predictive analytics while ensuring data privacy.
- Evaluate the practicality and efficiency of HE with ML algorithms in smart home data sets.

This research study aims to fulfil these objectives by analyzing how innovation in smart home technology can be balanced with strong privacy and security measures using advanced ML algorithms and encryption techniques. Section 2 highlights the related work in this field, and Section 3 will explore the background of smart home datasets, privacy and security threats, and a detailed explanation of HE. Section 4 provides a detailed account of the system's methodology and findings. Section 5 discusses the results of the study and its implications, and Section 6 concludes the article.

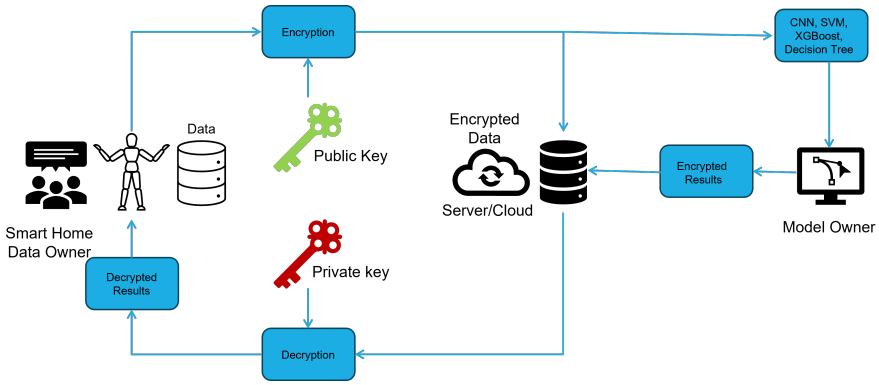

Figure 1. Smart home data with homomorphic encryption and machine learning depiction.

2. Literature Review

Privacy and security are essential when dealing with smart home or IoT sensor datasets, especially in the context of ML training [9,10]. By performing computations on encrypted data, one can create a robust encryption solution, thereby maintaining privacy and security. Thus, there is an extensive and diverse literature regarding HE and its applications in privacy-preserving data processing. This review summarizes the important studies conducted on the use and implementation of HE in various domains.

The HE model offers a promising solution by providing computations on encrypted data without the need to decrypt it first, thus maintaining data confidentiality throughout the process [11]. However, the SHE and PHE incur big efficiency and scalability drawbacks and may be prohibitive in large-scale applications such as smart home environments. To overcome these drawbacks, the possibility of using hybrid HE has been suggested. A combination of symmetric cryptography and HE achieves the same goals as HE but with increased performance and scalability [12]. The author proposed a Privacy-Preserving Machine Learning (PPML) scheme tailored for end devices, demonstrating minimal communication and computation costs while maintaining data privacy.

Another model [13], a hybrid HE-based PPML, was developed for the classification of heart disease with sensitive ECG data, showing only a slight accuracy reduction compared to plaintext. FHE was used to secure the computations and predictions in machine learning tasks such as predicting diabetes using logistic regression over encrypted datasets, which proved that during this process, data remained private and secure. Specifically, homomorphic evaluation of machine learning algorithms, deep Convolutional Neural Networks (CNNs), and Weightless Neural Networks (WNNs), has proven to be fairly practical in both inference and training on encrypted data, wherein high speedups have been achieved with competitive accuracy.

Similarly, HETAL [14] is an efficient HE-based transfer learning algorithm that focuses on protecting client privacy during training tasks by encrypting client data using the CKKS HE scheme. The study introduces an efficient encrypted matrix multiplication algorithm that is significantly faster, as well as a highly precise softmax approximation algorithm. Thus, the integration of Federated Learning (FL) with HE in the context of smart home sensor datasets provides a robust solution for privacy and security [10]. By utilizing AI models and the CKKS algorithm, FL enables collaborative training without sharing raw data, while HE ensures secure computation on encrypted data. This approach guarantees privacy preservation and accurate results, as demonstrated by achieving the highest average accuracy.

In ref. [15], the Multi-Key HE Logistic Regression (MK-HELR) algorithm is introduced. It lets you use shared datasets to do logistic regression on encrypted data from multiple parties while keeping the data private. In Ref. [16], FHE has been applied even to more complex models in machine learning, like CNNs, and has seen huge improvements in efficiency and accuracy. When comparing convolutions, optimized FHE methods cut the time needed by up to 46%. This implies that deep learning tasks such as CIFAR-

10/100 and ImageNet can utilize FHE. Better models, such as the ResNet-20 model with RNS-CKKS FHE [17], can produce results that are very similar to those of models that are not encrypted. For example, the ResNet-20 model with RNS-CKKS FHE obtained a score of 92.43% for CIFAR-10. This opens up new ways to use FHE with deep learning models in smart home environments. Advances in HE have huge potential for securing smart home sensor datasets while enabling sophisticated machine learning applications. In Table 1, one can see a detailed overview of all approaches, with additional methods included for a more comprehensive comparison. The table outlines each approach's key characteristics, encryption method used, and limitations, providing a clear understanding of how they perform.

These developments demonstrate how HE and its variants can make a difference in securing IoT datasets for safe and efficient machine learning applications that retain user privacy. Machine learning algorithms can easily integrate it, allowing for data analysis and prediction without compromising data privacy. The integration of HE and machine learning opens a very promising approach to dealing with privacy and security issues in smart home sensor datasets, opening a pathway toward more advanced data mining techniques securely and confidentially.

Table 1. Comparison of HE-based approaches in IoT-based datasets.

Paper Title	Encryption Method	Key Results	Limitations
Lattice-Based HE for Privacy-Preserving [18]	Lattice-based fully homomorphic encryption	Efficient smart meter data encryption and aggregation; allows computations on encrypted data	Key management complexity; potential inefficiency in large-scale deployments
Homomorphic Consortium Blockchain for Smart Home System [19]	HE in a consortium blockchain with multi-blockchain structure	Efficient data privacy preservation in smart home systems with distributed consensus on encrypted data	Complex key management and potential scalability issues in large networks
Smart Grids + Paillier HE [20]	Paillier homomorphic encryption for smart grid data Aggregation	Improved data privacy and encryption efficiency in smart grids; reduced encryption time compared to other methods	High computational complexity and potential performance bottlenecks in large-scale smart grid data
Lattice-Based and HE in Smart Grid [21]	Lattice-based homomorphic encryption for smart grid data aggregation	Enhanced consumer privacy and data confidentiality in smart grids; resilient to several attacks	Requires low computational overhead but may be vulnerable to lattice-based attacks
GuardML [12]	Hybrid Homomorphic Encryption (HHE) with PASTA for PPML in constrained devices	Demonstrated practicality of HHE in MLaaS with minimal accuracy loss; low computation and communication overhead	Slight reduction in model accuracy compared to plaintext; potential scalability challenges in large deployments
HETAL [14]	CKKS HE scheme, optimized for transfer learning by encrypting client data before training	Achieves accuracy near non-encrypted training; optimized matrix multiplication improves performance significantly	High computational overhead, especially for encrypted matrix multiplication
Privacy-Preserving ML With FHE [17]	FHE for privacy-preserving ML with function approximation	Achieved function approximation on encrypted data, balancing accuracy and privacy	Limited to small-scale models; performance degrades with increased model complexity

Table 1. *Cont.*

Paper Title	Encryption Method	Key Results	Limitations
Federated Learning and HE [10]	CKKS (Cheon–Kim–Kim–Song) algorithm integrated with federated learning	Achieved highest accuracy of 97.3% while maintaining data privacy in distributed ML environments	High computational overhead, with significant encryption and aggregation time
Lattice-based HE [18]	Lattice-based fully homomorphic encryption	Efficient smart meter data encryption and aggregation; allows computations on encrypted data	Key management complexity; potential inefficiency in large-scale deployments
Homomorphic WiSARDs [13]	TFHE (Fast Fully Homomorphic Encryption) for WiSARD model	Demonstrated effective encrypted RAM unit generation with lookup tables	High computational requirements for large datasets and complex operations
Random Forest using Multi-Key HE [22]	Multi-Key Homomorphic Encryption (MKHE) for Random Forest classification	Achieved 97.0% precision with encrypted Random Forest models; minimal performance degradation	Not suitable for large models; significant classification time (15.3s per feature vector)

3. Privacy and Security Threats in IoT Domain

Promoting independence and safety while monitoring the health and well-being of the elderly has recently emerged as a serious concern. This is the reason why, in the current period, smart home environments come with a wide range of sensors and use different embedded devices, such as Jetsons, to better support and monitor the daily routine and activities of the elderly. Therefore, these embedded devices gather large amounts of personal data, identifying activities, detecting health or behavioral abnormalities, and providing support through automated response or voice assistance. However, this extensive data collection raises serious concerns about user privacy. HE architecture is one of the state-of-art models that offers a promising solution by enabling computations on encrypted data, thus ensuring privacy and security while allowing meaningful data analysis.

3.1. Homomorphic Encryption

HE model, as proposed [23], is a type of encryption that facilitates performing computations on ciphertext. The final encrypted result, once decrypted, is identical to the plaintext result. It is useful for applications, especially IoT datasets, where data privacy is an issue, such as in smart home environments [19] where sensitive data are being generated and stored frequently. HE is classified into three different forms based on the complexity of the operations it supports:

- Partially Homomorphic Encryption (PHE): Supports either addition or multiplication but not both.
- Somewhat Homomorphic Encryption (SHE): Supports limited addition and multiplication operations.
- Fully Homomorphic Encryption (FHE): Supports arbitrary addition and multiplication operations, enabling any computation on encrypted data.

3.1.1. Homomorphic Encryption Schemes

HE schemes describe cryptographic protocols for performing computation on encrypted data without the need to first decrypt it. This promising approach maintains data privacy and confidentiality while enabling meaningful operations, making it highly useful in secure data processing, cloud computing, and machine learning that preserves privacy. The most prevalent of the many HE schemes are the BFV [24], BGV [25], and CKKS [26] schemes. These schemes vary depending on the technique used to manage arithmetic operations over encrypted data and handle noise in a way that balances precision with

computational efficiency. While both the BGV and BFV schemes perform exact arithmetic on encrypted integers, the former offers greater flexibility in managing noise. On the other hand, CKKS aims to approximate real and complex arithmetic, catering to applications where a slight loss of precision can significantly enhance performance. Table 2 provides a detailed comparison of HE approaches.

Table 2. Type of homomorphic encryption.

Scheme	Operations Supported	HE Tpye	Key Features	Security Basis	Use Case
BFV [24] (Brakerski Fan Vercauteren)	Addition, Multiplication	FHE	Supports exact arithmetic on integers, ciphertext packing	Ring Learning With Errors (RLWE)	Privacy-preserving computations on encrypted databases
BGV [25] (Brakerski Gentry Vaikuntana)	Addition, Multiplication	FHE	Flexible noise management, deeper circuit support	Ring Learning With Errors (RLWE)	Complex computations with deeper circuits
CKKS [26] (Cheon-Kim-Kim-Song)	Addition, Multiplication	FHE	Supports approximate arithmetic or real/complex numbers	Ring Learning with Errors (RLWE)	Machine learning, data analysis
Paillier [27] Encryption	Addition	PHE	Efficient additive homomorphism, public-key scheme	Decisional Composite Residuosity Assumption (DCRA)	Secure voting, privacy-preserving aggregation
FHEW [28] (Fast Homomorphic Encryption)	Addition, Multiplication	FHE	Efficient gate-by-gate evaluation, fast bootstrapping	Lattice-based	Secure circuit evaluation, Boolean operations
TFHE [29] (Torus Fully Homomorphic Encryption)	Addition, Multiplication	FHE	Fast binary gate operations, efficient bootstrapping	Lattice-based, torus operations	Private information retrieval, secure MPC

3.1.2. Primarily Study: HE Model Exploration

HE model is widely used in the ML domain. Therefore, there are multiple libraries available that support different forms of HE. These libraries differ in their underlying encryption schemes, ease of use, and computational efficiency. Some of the most common libraries are as follows:

- SEAL (Simple Encrypted Arithmetic Library) [30]: Microsoft® developed the SEA. It supports both BFV and CKKS schemes, which fit very well with the encrypted arithmetic of integers and real numbers, respectively.
- HElib [31]: IBM developed HElib, which is open source. It implements the BGV scheme in C++ and supports efficient arithmetic on encrypted data.
- TenSEAL [32]: A library for HE-based encrypted tensor manipulation, using Microsoft SEAL as its base. It is developed to integrate seamlessly with machine learning frameworks like PyTorch.
- Zama's Concrete-ML [33]: Yet another emerging, highly promising solution. It is based on Torus Fully Homomorphic Encryption (TFHE), the FHE framework on which Concrete-Ml develops. TFHE operates directly on encrypted bits and allows for Boolean operations with quite promising efficiency and security.

- OpenFHE [32] (previously known as PALISADE): This supports all major FHE schemes, including BGV, BFV, CKKS, DM (FHEW), and CGGI (TFHE) schemes.

As discussed previously, the TFHE is based on the General Learning with Errors (GLWE) problem, an extension of the Learning With Errors (LWE) framework, which itself forms the backbone of modern cryptographic solutions. The GLWE framework enhances the robustness of the encryption scheme by allowing for a broader range of error distributions, thus offering enhanced security against a wide spectrum of cryptographic attacks. This foundation enables TFHE to perform fast bootstrapping, a process to refresh the noise in ciphertexts, allowing for an unlimited number of computations without decrypting the data.

3.2. Privacy Attacks

Smart home systems, which collect and process data through different sensors and devices, present considerable possibilities for enhancing home management and security. However, they also expose individuals to a wide range of serious privacy and security vulnerabilities. Therefore, developing more secure systems requires a thorough understanding of these vulnerabilities and potential privacy attacks.

- The devices typically gather sensitive information about user behaviors, routines, and preferences. Improper protection or sharing of data can lead to privacy leakage.
- Most devices store and process information in cloud services. If these cloud assistants were insecure, it would be simple to access the stored information or even modify the device configuration.
- Attackers could monitor smart home devices' data to track users' habits, routines, and lifestyles, targeting them with stalking and burglary advertisements. This could help an attacker intercept communication on smart home devices to steal or manipulate sensitive data.

3.2.1. Attack Prevention Using HE with ML

ML algorithms, specifically classification algorithms, play a crucial role in analyzing smart home datasets. Libraries like Concrete-ML [33] have expanded support for integrating HE with selected ML models. This integration confirms that models can be trained and evaluated on encrypted data, maintaining the privacy of the underlying information. For this study, we have used three popular classification algorithms: Decision Classifier, XGBoost, and Random Forest.

- Decision Classifier [34]: For classification tasks, the Decision Classifier is a simple yet powerful algorithm. It constructs a decision tree based on the data features, making decisions at each node to classify the data into different categories.
- XGBoost [35]: This optimized gradient boosting method is very popular for its high performance and accuracy. In XGBoost, an additive collection of trees performs the boosting, with each tree correcting the error of its predecessor. This makes it very effective for a wide variety of machine-learning tasks.
- Random Forest [36]: This is a popular variant of ensemble learning in the domains of classification, regression, and anomaly detection. The technique's logic is that many decision trees are constructed, allowing the learner to improve the accuracy of their predictions and avoid overfitting. The Random Forest constructs each tree based on a unique bootstrap sample from the original data, ensuring a robust and reliable feature space in the problem.

3.2.2. Rationale for Choosing Tree-Based Models

In this study, we selected tree-based models—specifically XGBoost, Random Forest, and Decision Classifier—due to their robustness and interpretability, which are crucial for the following reasons:

- Handling Mixed Data Types: Tree-based models are well-suited for datasets with a mix of numerical and categorical features. Given the nature of our smart home dataset,

which includes diverse types of sensor data, tree-based models handle these mixed types effectively without the need for extensive preprocessing.

- Model Interpretability: Tree-based models, particularly Decision Trees and Random Forests, provide more interpretability compared to complex deep learning approaches. This interpretability is valuable for understanding how different features contribute to the model's predictions, which is important for analyzing and validating results in practical applications.
- Performance with Encrypted Data: Tree-based models are known to perform well with encryption techniques, including Fully Homomorphic Encryption (FHE). Their structure allows for efficient handling of encrypted data, which is crucial for maintaining both security and performance in our study.
- Computational Efficiency: Compared to deep learning models, tree-based models generally require less computational power and training time, making them more practical for the scope of our study, especially when evaluating performance with FHE.

We considered alternatives such as MLP, SVM, and deep learning approaches but determined that the advantages of tree-based models in terms of handling mixed data types, interpretability, and computational efficiency aligned better with the objectives of our research.

4. Methodology

The methodology for this study involves several key steps, each aimed at ensuring the secure processing of data in smart home environments while maintaining high accuracy in activity recognition. The steps are as follows.

4.1. Data

The dataset is taken from CASAS [37] (HH101), single apartment resident data from various sensors and smart devices using edge devices. This dataset consists of continuously collected ambient data from homes inhabited by volunteer residents as they perform their daily routines. These include data from motion sensors, door sensors, smart meters, and other devices that can provide insights into the activities of residents. Figure 2 shows the sensor's location in the apartment. Each sensor's position is identified by its subsequent type, Motion (M), Motion Area (MA), Light (LS), Door (D), Temperature (T), with sensor ID shown in Table 3.

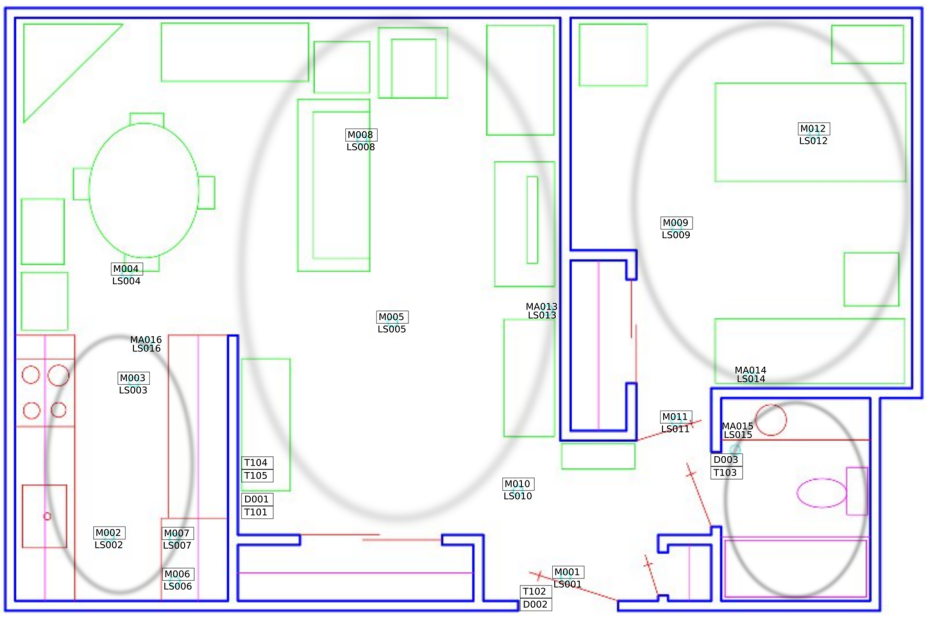

Figure 2. Location of sensors in smart home environment [38].

Table 3. Description of sensor placement in HH101 dataset.

Sensor Type	Sensor ID	Sensor Location
M, LS	001	Entry
M, LS	002	Kitchen
M, LS	003	Kitchen
M, LS	004	Dinning Area
M, LS	005	Living Room
M, LS	006	Kitchen
M, LS	007	Kitchen
M, LS	008	Living Room
M, LS	009	Bedroom
M, LS	010	Hallway
M, LS	011	Bedroom
M, LS	012	Bedroom
MA, LS	013	Living Room
MA, LS	014	Bedroom
MA, LS	015	Toilet
MA, LS	016	Kitchen
D, T	01	Living
D, T	02	Main Entry
D, T	03	Toilet

4.2. Data Encryption: Pre-Processing

For FHE, we utilized the Concrete-ML library, an open-source, privacy-preserving machine learning framework based on FHE. This library was selected for its robustness and efficiency in handling FHE operations on real-world datasets. The first step in our proposed approach is to preprocess the data before training a ML model on plaintext data using a library such as Scikit-learn.

4.2.1. Handling Missing Values

To ensure the integrity of the dataset, missing values were handled using mean imputation, where any missing data points were replaced with the mean value of the corresponding feature. This approach helps maintain the dataset's structure without introducing significant biases [39,40].

4.2.2. Data Normalization

Given that FHE operates on integers, data normalization was a critical step before applying encryption. We used min–max scaling to transform the feature values into a fixed range, typically [0,1], ensuring uniformity across features. This transformation is essential because FHE requires integer-based computations, and normalization helps maintain model performance during encryption and inference [41,42].

4.2.3. Model Quantization

Concrete-ML quantizes the trained model, converting it into an integer-only format for inference. This quantization process is crucial for compatibility with FHE, ensuring that the encrypted data remain secure while allowing for accurate model predictions [43].

4.2.4. Compilation and Encryption

Once the model is quantized, it is converted into a Concrete-ML-based execution process and compiled. This step is considered part of the pre-processing phase as it prepares the model for efficient execution on encrypted data [44]. Finally, inference is performed directly on encrypted data, ensuring privacy without compromising the model's predictive capabilities.

4.3. Model Training–Processing

To ensure that the model's performance was validated and generalizable, we split the dataset into training and test sets using the train_test_split function. Specifically, we used 70% of the data for training and reserved 30% for testing. The training data were used to train the Decision Classifier, while the test data were kept separate to evaluate the model's performance on unseen data. By keeping the test set isolated from the training process, we ensured that the evaluation was unbiased and reflected the model's ability to generalize. We then trained and concluded the chosen model based on the dataset. We trained models such as XGBoost, Random Forest, and Decision Classifiers on a dataset before conducting inference on an encrypted dataset. While testing models on encrypted data is significantly more complex than on plaintext data, it guarantees the privacy of the raw data by never revealing it.

4.4. Analysis

Finally, the encrypted results are analyzed. The accuracy and execution time of the models were evaluated to determine the effectiveness of FHE in maintaining data privacy without significantly compromising performance. The algorithms for models without and with FHE are shown in Algorithms 1 and 2.

Algorithm 1 Model (Plaintext)

1: Initialize the `Model`.
2: Record start time using `time.time()`.
3: Fit the model on the training data $X_{\text{train}}, y_{\text{train}}$.
4: Predict on the test data X_{test}.
5: Calculate the total duration for training and prediction.
6: Calculate accuracy using the predicted and actual test labels.
7: Print the total duration and model accuracy.
8: Generate a classification report for the predictions.

Algorithm 2 Model with Fully FHE

1: Initialize the `Model` with FHE enabled (`n_bits=8`)
2: Fit the model on the training data $X_{\text{train}}, y_{\text{train}}$
3: Compile the model for FHE execution
4: Select a subset of the test data $X_{\text{test\_fhe}}, y_{\text{test\_fhe}}$ for FHE execution
5: Record start time using `time.time()`
6: Simulate FHE prediction on the selected test data
7: Execute FHE prediction on the selected test data
8: Calculate the total duration for FHE training and prediction
9: Generate a classification report for the FHE predictions
10: Calculate accuracy using the predicted and actual FHE test labels
11: Print the total duration and model accuracy with FHE

5. Results and Discussion

The dataset includes various sensor readings and activities recorded over a period of time that capture the daily routines and behaviors of the resident. The dataset has 35 unique attributes and 321,428 tuples. The total number of activities in the dataset is plotted in Figure 3.

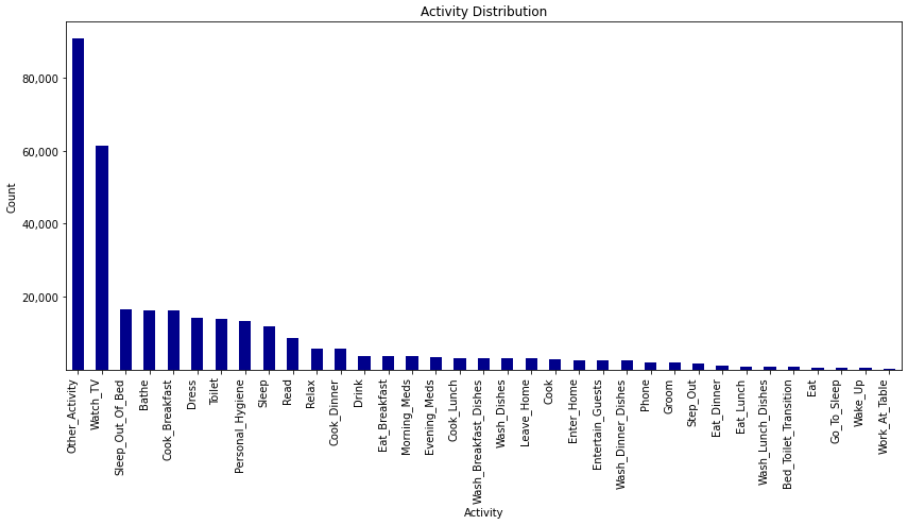

Figure 3. Number of activities in a dataset.

5.1. Models Performance

Model performance is evaluated and compared with accuracy, precision, recall, f1-score, RMSE, hamming loss, and inference time. Accuracy refers to the ratio of correctly predicted instances, that is, both the true positives and the true negatives, against the total number of instances in the dataset. It provides an overall measure of how often the model is correct.

$$\text{Accuracy} = \frac{\text{True Positives + True Negatives}}{\text{Total Number of Instances}} \tag{1}$$

Precision is the proportion of rightly predicted positive values out of all the records that are predicted as positive.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}} \tag{2}$$

Recall is the ratio of the number of correctly predicted positive records to the total actual positive records. It is also known as the True Positive Rate. The F1 score gives one single metric, which considers both precision and recall when you need to maintain a balance between these two.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

RMSE (Root Mean Square Error) is used to measure the accuracy of the models in predicting continuous outcomes; the lower the values, the better.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum(y_i' - y_i)^2}, \tag{4}$$

where n is the number of samples, y' is the predicted value, and y is the actual value.

In a multi-label classification task, the hamming loss measures the fraction of incorrectly predicted labels and provides insight into how accurate the models are in classification scenarios.

Inference time is the time taken to evaluate each model's computational efficiency during prediction. The results indicate a significant drop in accuracy when FHE is applied. This can be attributed to the additional noise introduced during the encryption process, which affects the performance of the ML models. Additionally, the execution time increased dramatically with FHE, highlighting the computational overhead associated with processing encrypted data.

- The accuracy score with and without FHE for all three models is shown in Table 4. The accuracy for XGBoost and Random Forest on plaintext is almost the same, except for the Decision Classifier, which has 96% accuracy. Encrypting the test data and drawing inferences from them results in a minimal drop in accuracy compared to models without encryption. However, the Decision Classifier performs better with FHE than both plaintext and the other two models.
- In the cases of precision, recall, and F1 score, we identified a similar trend. When applying FHE, the Decision Classifier model demonstrates remarkable resilience and makes a significant difference in maintaining precision and recall. For Random Forest and XGBoost, encrypting data results in degradations primarily in precision and recall, as shown in Figure 4. The F1 score, which balances precision and recall, reflected this minor degradation. On average, the Decision Classifier performed best in maintaining performance metrics using FHE, making it an ideal candidate for privacy-preserving machine learning tasks.
- To determine the computational overhead introduced by encryption, we made a comparison of the execution times for models with and without FHE, as depicted in Figure 5. In XGBoost, execution time increased slightly with FHE, as anticipated due to the additional computational overhead associated with encryption. Random Forest's inference time is almost negligent, indicating a possible improvement in encrypted data handling. The Decision Classifier has a significantly higher execution time under FHE, showing the overhead added by encryption. It may therefore not be appropriate for applications that require real-time performance. The analysis suggests that even though FHE adds overhead, model selection could critically impact this; XGBoost was the most efficient model relative to execution time.
- To determine the percentage of incorrect predictions, we calculate the hamming loss. As shown in Figure 6, XGBoost recorded the lowest hamming loss without FHE, indicating excellent predictive performance in plaintext. The Decision Classifier exhibited a noteworthy reduction in loss with FHE, ranking lowest among all three models. This enabled it to sustain a balanced performance during encryption. However, without FHE, its accuracy declined. Table 5 shows a detailed comparison of the results.
- In the case of XGBoost and Random Forest without FHE, the RMSE is lower, showing high predictive accuracy. In a Random Forest with FHE, the RMSE increased, resulting in less accuracy under encryption. As shown in Figure 7, the RMSE of the Decision Classifier with FHE is lower than that of the Random Forest and XGBoost classifiers.

Table 4. Performance comparison of models with and without FHE.

	XGBoost	XGBoost with FHE	Random Forest	Random Forest with FHE	Decision Classifier	Decision Classifier with FHE
Accuracy	98.00%	94.40%	98.35%	92.20%	96.73%	98.00%
Precision	98.00%	94.00%	98.00%	92.00%	97.00%	98.00%
Recall	98.00%	94.00%	98.00%	92.00%	97.00%	98.00%
F1 Score	98.00%	94.00%	98.00%	92.00%	97.00%	98.00%

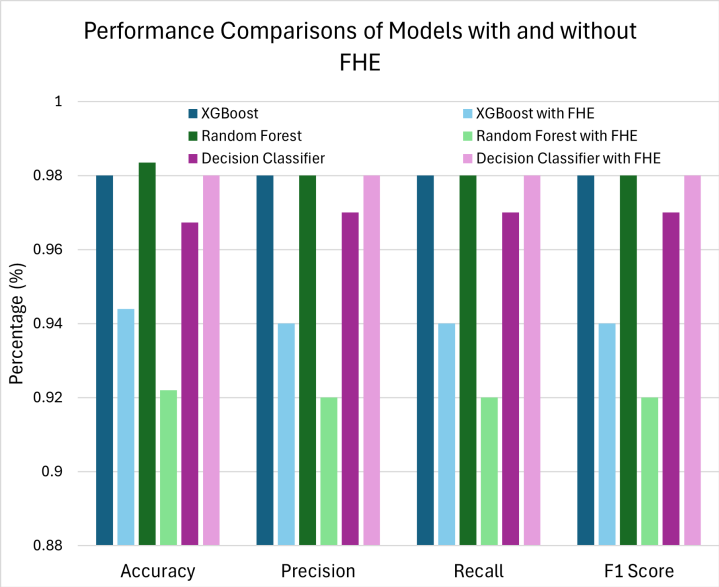

Figure 4. Accuracy, precision, recall, and F1 Score comparison of XGBoost, Decision Classifier, and Random Forest with FHE and without encryption.

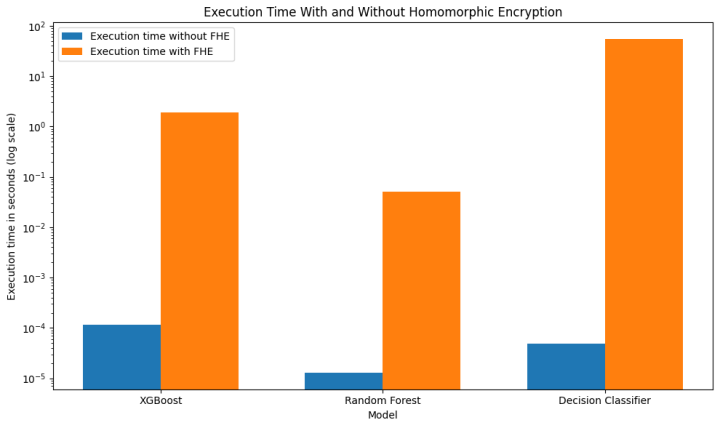

Figure 5. Execution Time of XGBoost, Random Forest, and Decision Classifier with and without FHE.

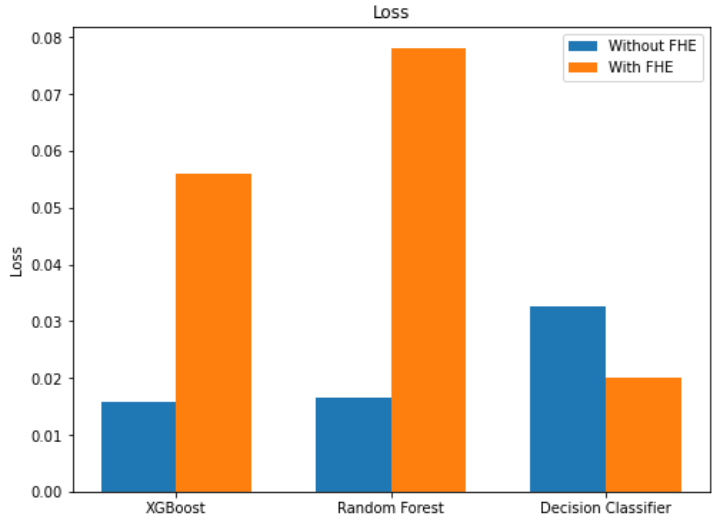

Figure 6. Loss of XGBoost, Random Forest, and Decision Classifier with and without FHE.

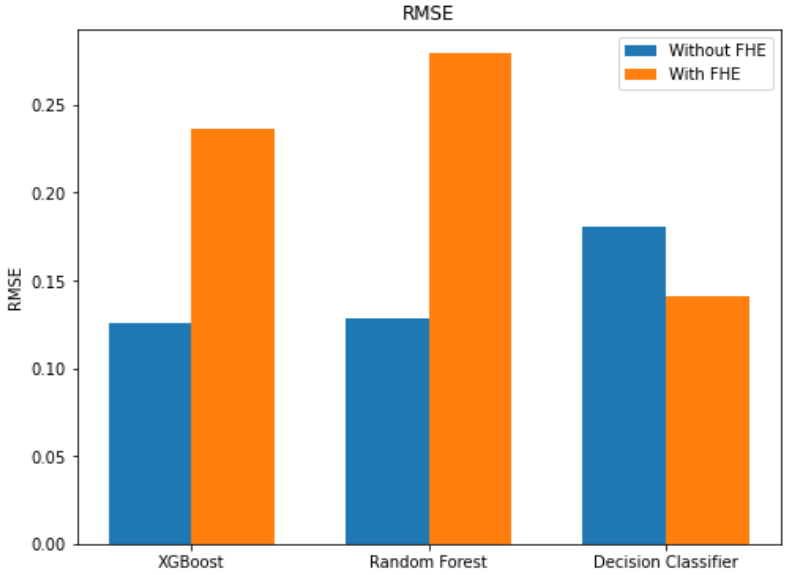

Figure 7. RMSE of XGBoost, Random Forest, and Decision Classifier with and without FHE.

Table 5. Execution time, loss, and RMSE in XGBoost, Random Forest, and Decision Classifier with and without FHE.

	XGBoost	XGBoost with FHE	Random Forest	Random Forest with FHE	Decision Classifier	Decision Classifier with FHE
Execution Time (s)	0.000117	1.90000	0.000013	0.0500	0.000049	54.170
Loss	0.015889	0.05600	0.016500	0.0780	0.032690	0.0200
RMSE	0.126051	0.23664	0.128409	0.2792	0.180801	0.1414

5.2. Discussion

The proposed study highlights that the introduction of small noise by FHE enhances the accuracy of the Decision Classifier when compared to plaintext data. Without this noise, the model's sensitivity to outliers or noisy instances in the data could have been too strong. The encryption process may add small, consistent noise across similar data points, effectively reducing the noise and making the data more homogeneous. This kind of homogenization may help the Decision Classifier come up with more consistent and generalized rules. Although FHE introduces some computational overhead and loss in model performance, choosing the right ML model can effectively manage this. XGBoost has been the best performer on all metrics for accuracy, execution time, RMSE, and hamming loss, thus increasing its popularity for the application that focuses on privacy preservation using FHE. On the other hand, models such as Random Forest are efficient in terms of time, and Decision Classifiers performed well on encrypted data in terms of accuracy and precision. By balancing security and performance, the analysis has shed some important light on the choice of models for encrypted environments.

Scalability and Feasibility of Fully Homomorphic Encryption

FHE offers a strong privacy guarantee by allowing computations on encrypted data without the need for decryption. However, its primary challenge lies in the computational overhead, which limits its scalability in real-world applications, such as smart home activity recognition. In this study, we applied FHE uniformly across the entire dataset to ensure consistent privacy protection. Selectively applying FHE to only sensitive data, as seen

in hybrid encryption approaches, might reduce computational load but is difficult to implement in smart home environments. Since even seemingly non-sensitive data can reveal critical patterns of behavior, leaving parts of the data unencrypted under a less secure scheme would introduce privacy risks.

To mitigate the performance issues associated with FHE, several scalability solutions have been explored in recent work, and these can be leveraged to improve the feasibility of FHE in practice:

- Cloud-based Parallelization: By distributing the FHE computations across cloud platforms with multi-core or GPU-based systems, the processing time can be significantly reduced. This allows for the practical application of FHE at scale.
- Efficient FHE Libraries: New optimizations in FHE libraries (e.g., CKKS, TFHE) focus on improving bootstrapping times, which is traditionally a bottleneck in FHE computations. These optimizations make FHE more feasible for larger datasets.
- Approximate Homomorphic Encryption: While our study focused on exact FHE methods, approximate homomorphic encryption could also be explored in future work, where slight imprecisions in data are acceptable, thus reducing computational complexity.

6. Conclusions

FHE offers a powerful solution to the challenges of privacy and security in data processing within smart home environments. By enabling computations on encrypted data, FHE ensures that sensitive information remains fully protected while still allowing valuable insights and functionalities to be derived. In this paper, we compared the performance of three machine learning models, XGBoost, Random Forest, and Decision Classifier, on a smart home dataset, with and without FHE. The results revealed that the Decision Classifier maintains high accuracy with minimal degradation when FHE is applied, demonstrating its robustness even with the added encryption overhead. Interestingly, the Decision Classifier with FHE not only performed better than the other models but also surpassed its own accuracy in plaintext, highlighting the potential for encryption to enhance model performance under certain conditions. These findings underscore the efficiency of Random Forest in terms of execution time and low prediction errors when handling FHE, making it a strong candidate for encrypted data processing in smart homes. This study demonstrates that FHE can indeed be transformative in setting high privacy standards in smart homes, paving the way for more secure and privacy-preserving technologies.

Future research can explore the integration of HE with federated learning, as well as the combination of FHE with emerging technologies like blockchain and edge computing, to further enhance the security and privacy of smart home environments. Continued advances in encryption techniques and machine learning algorithms will be crucial to fully utilizing the potential of privacy-preserving smart home systems.

Author Contributions: H.A. led the study design, implemented machine learning models, conducted primary experiments, and performed comparative analysis. S.S. assisted with the implementation, literature review, background research, and drafting of the initial sections of the manuscript. T.J. provided expert guidance on encryption techniques, supervised the entire project, and reviewed and refined the manuscript before submission. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by the research group "Transformation in Care and Technology TransCareTech". This project is funded by the Ministry of Culture and Science of the State of North Rhine-Westphalia, and the grant number is PROFILNRW-2020-095.

Data Availability Statement: Datasets are available online at http://casas.wsu.edu (accessed on 26 August 2024).

Acknowledgments: This research is also supported by the research training group "Dataninja" (Trustworthy AI for Seamless Problem Solving: Next Generation Intelligence Joins Robust Data Analysis), funded by the German federal state of North Rhine-Westphalia.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

HE	Homomorphic Encryption
ML	Machine Learning
IoT	Internet of Things
FHE	Fully Homomorphic Encryption

References

1. Chataut, R.; Phoummalayvane, A.; Akl, R. Unleashing the power of IoT: A comprehensive review of IoT applications and future prospects in healthcare, agriculture, smart homes, smart cities, and industry 4.0. *Sensors* **2023**, *23*, 7194. [CrossRef] [PubMed]
2. Asaithambi, S.P.R.; Venkatraman, S.; Venkatraman, R. Big data and personalisation for non-intrusive smart home automation. *Big Data Cogn. Comput.* **2021**, *5*, 6. [CrossRef]
3. Sepasgozar, S.; Karimi, R.; Farahzadi, L.; Moezzi, F.; Shirowzhan, S.; Ebrahimzadeh, S.M.; Hui, F.; Aye, L. A systematic content review of artificial intelligence and the internet of things applications in smart home. *Appl. Sci.* **2020**, *10*, 3074. [CrossRef]
4. Sanaullah; Baig, H.; Madsen, J.; Lee, J.A. A parallel approach to perform threshold value and propagation delay analyses of genetic logic circuit models. *ACS Synth. Biol.* **2020**, *9*, 3422–3428. [CrossRef] [PubMed]
5. Elmustafa, S.A.A.; Mujtaba, E.Y. Internet of things in smart environment: Concept, applications, challenges, and future directions. *World Sci. News* **2019**, *134*, 1–51.
6. Sanaullah, S. A Hybrid Spiking-Convolutional Neural Network Approach for Advancing Machine Learning Models. In Proceedings of the Northern Lights Deep Learning Conference, PMLR, Tromsø, Norway, 9–11 January 2024; pp. 220–227.
7. Jungeblut, T. A Spike Vision Approach for Multi-object Detection and Generating Dataset Using Multi-core Architecture on Edge Device. In Proceedings of the Engineering Applications of Neural Networks: 25th International Conference, EANN 2024, Corfu, Greece, 27–30 June 2024; Springer: Amsterdam, The Netherlands, 2024; p. 317.
8. Zama. TFHE-rs: A Pure Rust Implementation of the TFHE Scheme for Boolean and Integer Arithmetics Over Encrypted Data. 2022. Available online: https://github.com/zama-ai/tfhe-rs (accessed on 26 August 2024).
9. Sanaullah, S.; Attaullah, H.; Jungeblut, T. The Next-Gen Interactive Runtime Simulator for Neural Network Programming. In Proceedings of the 8th International Conference on the Art, Science, and Engineering of Programming, Lund, Sweden, 11–15 March 2024; pp. 8–10.
10. Nguyen, T. Advancing Privacy and Accuracy with Federated Learning and Homomorphic Encryption. *Authorea Prepr.* **2023**. [CrossRef]
11. Krishna, N.; Raju, K.M.; Gowda, V.D.; Arun, G.; Suneetha, S. Homomorphic Encryption and Machine Learning in the Encrypted Domain. In *Innovative Machine Learning Applications for Cryptography*; IGI Global: Hershey, PA, USA, 2024; pp. 173–190.
12. Frimpong, E.; Nguyen, K.; Budzys, M.; Khan, T.; Michalas, A. GuardML: Efficient Privacy-Preserving Machine Learning Services Through Hybrid Homomorphic Encryption. In Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing, Avila, Spain, 8–12 April 2024; pp. 953–962.
13. Neumann, L.; Guimarães, A.; Aranha, D.F.; Borin, E. Homomorphic WiSARDs: Efficient Weightless Neural Network training over encrypted data. *arXiv* **2024**, arXiv:2403.20190.
14. Lee, S.; Lee, G.; Kim, J.W.; Shin, J.; Lee, M.K. HETAL: Efficient privacy-preserving transfer learning with homomorphic encryption. In Proceedings of the International Conference on Machine Learning, PMLR, Honolulu, HI, USA, 23–29 July 2023; pp. 19010–19035.
15. Kang, D.H.E.; Kim, D.; Song, Y.; Lee, D.; Kwak, H.; Anthony, B.W. Harnessing the potential of shared data in a secure, inclusive, and resilient manner via multi-key homomorphic encryption. *Sci. Rep.* **2024**, *14*, 13626. [CrossRef]
16. Kim, D.; Guyot, C. Optimized privacy-preserving cnn inference with fully homomorphic encryption. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 2175–2187. [CrossRef]
17. Lee, J.W.; Kang, H.; Lee, Y.; Choi, W.; Eom, J.; Deryabin, M.; Lee, E.; Lee, J.; Yoo, D.; Kim, Y.S.; et al. Privacy-preserving machine learning with fully homomorphic encryption for deep neural network. *IEEE Access* **2022**, *10*, 30039–30054. [CrossRef]
18. Marandi, A.; Alves, P.G.M.; Aranha, D.F.; Jacobsen, R.H. Lattice-Based Homomorphic Encryption For Privacy-Preserving Smart Meter Data Analytics. *Comput. J.* **2023**, 67, 1687–1698. [CrossRef]
19. She, W.; Gu, Z.H.; Lyu, X.K.; Liu, Q.; Tian, Z.; Liu, W. Homomorphic consortium blockchain for smart home system sensitive data privacy preserving. *IEEE Access* **2019**, *7*, 62058–62070. [CrossRef]
20. Zhao, S. Smart Grids Data Aggregation Method on Paillier Homomorphic Encryption. *Appl. Artif. Intell.* **2024**, *38*, 2327901. [CrossRef]
21. Romdhane, R.B.; Hammami, H.; Hamdi, M.; Kim, T.H. At the cross roads of lattice-based and homomorphic encryption to secure data aggregation in smart grid. In Proceedings of the 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 24–28 June 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1067–1072.

22. Petrean, D.E.; Potolea, R. Random forest evaluation using multi-key homomorphic encryption and lookup tables. *Int. J. Inf. Secur.* **2024**, *23*, 2023–2041. [CrossRef]
23. Gentry, C. *A Fully Homomorphic Encryption Scheme*; Stanford University: Stanford, CA, USA, 2009.
24. Fan, J.; Vercauteren, F. Somewhat practical fully homomorphic encryption. *Cryptol. Eprint Arch.* **2012**, *2012*, 144.
25. Brakerski, Z.; Gentry, C.; Vaikuntanathan, V. (Leveled) fully homomorphic encryption without bootstrapping. *ACM Trans. Comput. Theory (TOCT)* **2014**, *6*, 1–36. [CrossRef]
26. Cheon, J.H.; Kim, A.; Kim, M.; Song, Y. Homomorphic encryption for arithmetic of approximate numbers. In Proceedings of the Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, 3–7 December 2017; Proceedings, Part I 23; Springer: Berlin/Heidelberg, Germany, 2017, pp. 409–437.
27. Paillier, P. Public-key cryptosystems based on composite degree residuosity classes. In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Prague, Czech Republic, 2–6 May 1999; Springer: Berlin/Heidelberg, Germany, 1999; pp. 223–238.
28. Ducas, L.; Micciancio, D. FHEW: Bootstrapping homomorphic encryption in less than a second. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 April 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 617–640.
29. Chillotti, I.; Gama, N.; Georgieva, M.; Izabachène, M. TFHE: Fast fully homomorphic encryption over the torus. *J. Cryptol.* **2020**, *33*, 34–91. [CrossRef]
30. Microsoft SEAL (release 4.1). Microsoft Research: Redmond, WA, USA, 2023. Available online: https://github.com/Microsoft/SEAL (accessed on 1 March 2024).
31. Halevi, S.; Shoup, V. Design and implementation of HElib: A homomorphic encryption library. *Cryptol. ePrint Archive* **2020**. Available online: https://github.com/homenc/HElib (accessed on 1 March 2024).
32. Badawi, A.A.; Alexandru, A.; Bates, J.; Bergamaschi, F.; Cousins, D.B.; Erabelli, S.; Genise, N.; Halevi, S.; Hunt, H.; Kim, A.; et al. OpenFHE: Open-Source Fully Homomorphic Encryption Library. Cryptology ePrint Archive, Paper 2022/915, 2022. Available online: https://eprint.iacr.org/2022/915 (accessed on 1 March 2024).
33. Zama. Concrete ML: A Privacy-Preserving Machine Learning Library Using Fully Homomorphic Encryption for Data Scientists, 2022. Available online: https://github.com/zama-ai/concrete-ml (accessed on 1 February 2024).
34. Du, W.; Zhan, Z. Building Decision Tree Classifier on Private Data. 2002. Available online: https://surface.syr.edu/cgi/viewcontent.cgi?article=1007&context=eecs (accessed on 1 March 2024).
35. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
36. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
37. University, W.S. CASAS Datasets for Smart Environments. 2024. Available online: http://casas.wsu.edu/datasets/ (accessed on 18 February 2024).
38. Cook, D.J.; Crandall, A.S.; Thomas, B.L.; Krishnan, N.C. CASAS: A smart home in a box. *Computer* **2012**, *46*, 62–69. [CrossRef]
39. Farhangfar, A.; Kurgan, L.A.; Pedrycz, W. A novel framework for imputation of missing values in databases. *IEEE Trans. Syst. Man Cybern. Part A Syst. Humans* **2007**, *37*, 692–709. [CrossRef]
40. Alabadla, M.; Sidi, F.; Ishak, I.; Ibrahim, H.; Affendey, L.S.; Ani, Z.C.; Jabar, M.A.; Bukar, U.A.; Devaraj, N.K.; Muda, A.S.; et al. Systematic review of using machine learning in imputing missing values. *IEEE Access* **2022**, *10*, 44483–44502. [CrossRef]
41. Adhikary, S.; Dutta, S.; Dwivedi, A.D. Secret learning for lung cancer diagnosis—A study with homomorphic encryption, texture analysis and deep learning. *Biomed. Phys. Eng. Express* **2023**, *10*, 015011. [CrossRef] [PubMed]
42. Salamatian, S. Statistical Privacy and Security. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2020.
43. Frery, J.; Stoian, A.; Bredehoft, R.; Montero, L.; Kherfallah, C.; Chevallier-Mames, B.; Meyre, A. Privacy-preserving tree-based inference with fully homomorphic encryption. *Cryptol. Eprint Arch.* **2023**, *preprint*
44. Xu, R.; Joshi, J.; Li, C. Nn-emd: Efficiently training neural networks using encrypted multi-sourced datasets. *IEEE Trans. Dependable Secur. Comput.* **2021**, *19*, 2807–2820. [CrossRef]