Check for updates

ACM DIGITAL LIBRARY | Association for Computing Machinery | acm open

PDF Download
3770633.pdf
22 December 2025
Total Citations: 0
Total Downloads: 153

Latest updates: https://dl.acm.org/doi/10.1145/3770633

Published: 28 November 2025

Citation in BibTeX format

RESEARCH-ARTICLE

Real-Time Encrypted Emotion Recognition Using Homomorphic Encryption

GYEONGWON CHA, Chung-Ang University, Seoul, South Korea

DONGJIN PARK, Chung-Ang University, Seoul, South Korea

YEJIN CHOI, Chung-Ang University, Seoul, South Korea

EUNJI PARK, Chung-Ang University, Seoul, South Korea

JOON-WOO LEE, Chung-Ang University, Seoul, South Korea

Open Access Support provided by:

Chung-Ang University

Real-Time Encrypted Emotion Recognition Using Homomorphic Encryption

GYEONGWON CHA*, Chung-Ang University, Republic of Korea
DONGJIN PARK*, Chung-Ang University, Republic of Korea
YEJIN CHOI, Chung-Ang University, Republic of Korea
EUNJI PARK†, Chung-Ang University, Republic of Korea
JOON-WOO LEE†, Chung-Ang University, Republic of Korea

Emotion recognition has been an actively researched topic in the field of HCI. However, multimodal datasets used for emotion recognition often contain sensitive personal information, such as physiological signals, facial images, and behavioral patterns, raising significant privacy concerns. In particular, the privacy issues become crucial in workplace settings because of the risks such as surveillance and unauthorized data usage caused by the misuse of collected datasets. To address this issue, we propose an Encrypted Emotion Recognition (EER) framework that performs real-time inference on encrypted data using the CKKS homomorphic encryption (HE) scheme. We evaluated the proposed framework using publicly available WESAD and Hide-and-seek datasets, demonstrating successful stress/emotion recognition under encryption. The results demonstrated that encrypted inference achieved similar accuracy to plaintext inference, with accuracy of 0.966 (plaintext) vs. 0.967 (ciphertext) on the WESAD dataset, and 0.868 for both cases on the Hide-and-Seek dataset. Encrypted inference was performed on a GPU, with average inference times of 333 milliseconds for the general model and 455 milliseconds for the personalized model. Furthermore, we validated the feasibility of semi-supervised learning and model personalization in encrypted environments, enhancing the framework's real-world applicability. Our findings suggest that the EER framework provides a scalable, privacy-preserving solution for emotion recognition in domains such as healthcare and workplace settings, where securing sensitive data is of critical importance.

Additional Key Words and Phrases: Data privacy, Emotion recognition, Cloud privacy, Privacy-preserving machine learning, Homomorphic encryption

ACM Reference Format:
Gyeongwon Cha, Dongjin Park, Yejin Choi, Eunji Park, and Joon-Woo Lee. 2025. Real-Time Encrypted Emotion Recognition Using Homomorphic Encryption. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 9, 4, Article 163 (December 2025), 32 pages. https://doi.org/10.1145/3770633

1 INTRODUCTION

Emotion recognition is an actively researched topic in the field of Human-Computer Interaction. Beyond estimating human stress and emotional status using physiological sensing data, recent studies have been focusing on utilizing multimodal datasets along with artificial intelligence models to classify emotions in real-time and provide

*Both authors contributed equally to this research.
†Co-corresponding authors.

Authors' Contact Information: Gyeongwon Cha, Chung-Ang University, Seoul, Republic of Korea, dbfldk20@cau.ac.kr; Dongjin Park, Chung-Ang University, Seoul, Republic of Korea, thrudgelmir@cau.ac.kr; Yejin Choi, Chung-Ang University, Seoul, Republic of Korea, yeyeye222@cau.ac.kr; Eunji Park, Chung-Ang University, Seoul, Republic of Korea, eunjipark@cau.ac.kr; Joon-Woo Lee, Chung-Ang University, Seoul, Republic of Korea, jwlee2815@cau.ac.kr.



This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.
© 2025 Copyright held by the owner/author(s).
ACM 2474-9567/2025/12-ART163
https://doi.org/10.1145/3770633

interventions. While using multimodal datasets in emotion recognition allows for the use of more extensive and diverse information, it also raises issues regarding the easier and more varied ways in which user privacy can be compromised. For example, physiological data can reveal a user's health and biological state, and images such as facial or iris images, which can identify individuals, carry the risk of personal information leakage. As the variety of collected data types increases and the volume of data grows, there is a risk of unintentionally inferring individual characteristics or lifestyle patterns that were not the original purpose of data collection.

The privacy issues of emotion recognition can be considered even more serious when applied in the context of workspaces. Recently, emotion recognition technology has been increasingly utilized in workplaces to enhance productivity and prevent worker emotional exhaustion [35, 48, 53, 65]. For this, the collection of private data such as the workplace environment (temperature, humidity, etc.) [53, 55, 76], the worker's status (physiological signals) [7, 25, 35, 53, 69], and interaction history between the worker and computer (desktop usage history, mobile device usage history [51, 84] have been required. While such research aims to improve worker well-being and productivity, it can conversely be used as a means of real-time automatic monitoring of worker productivity or directly influencing the hiring process. Additionally, if trust in employers is low, the very act of data collection can be perceived by workers as surveillance [57], potentially becoming a source of stress. Another privacy issue arising from the workspace context is customer privacy. In workplaces such as call centers for card or financial companies, calls with customers are recorded to protect workers from emotional labor and to detect the stress they may be experiencing. Since conversations between customers and workers can contain a significant amount of customer data, it is crucial to be particularly mindful of the collection and storage of voice data.

Privacy issues in emotion recognition are an area of active research within the field of HCI. However, until now, solutions have been limited to the use of federated learning [17, 22, 23, 28, 78], differential privacy techniques [38], or updating models on-device [26, 39, 75]. Federated learning protects personal information by localizing raw data and sending only the gradients or weights of the updated model based on that data to a global model. However, the information sent to the global model still contains data distribution information about users, which poses a risk of model inversion attacks or inference attacks that could trace back to raw data. Similarly, while differential privacy adds noise to the updates of gradients or weights sent to the global model, the accuracy of the model may decrease as more noise is included. Furthermore, since differential privacy is primarily designed to prevent the leakage of personal information during the training phase by ensuring that the trained model does not retain sensitive data, it is not applicable for guaranteeing data privacy during inference in real-time service settings. On-device learning, as another method, conducts training on local devices without sending data to a central server. This local processing is useful from a privacy perspective, but it is dependent on the performance of the device's hardware and software, making it relatively difficult to build a general model with good performance. One solution to these issues is the use of homomorphic encryption (HE). The HE is an advanced encryption technique designed to enable arithmetic operations directly on encrypted data, while satisfying the strong cryptographic security notion of IND-CPA (indistinguishability under chosen-plaintext attack). It encrypts raw data before sending it to the global model, allowing the model to be updated using the encrypted data. Since model training and inference can be performed based on the encrypted data, it offers the advantage of enhancing the performance of the global model while protecting individuals' private data. In this regard, we propose the following research questions:

- RQ1: How can HE be utilized to protect user privacy when using emotion recognition technology in the workspace?
- RQ2: What considerations should be made to enhance the practicality of emotion recognition models utilizing HE in real-world scenarios?

To address the first research question, we propose the Encrypted Emotion Recognition (EER) framework. This framework utilizes HE to encrypt the process of model training and inference based on private data collected in the workplace. Based on the framework, we validated whether the model provides sufficient inference

performance under encryption, while ensuring standard cryptographic security and supporting real-time service. Since performing computations on encrypted data using HE typically incurs a runtime overhead that is several orders of magnitude higher than plaintext operations, designing complex models under HE often makes real-time service infeasible. To address this, our model was carefully designed to be lightweight in structure and was efficiently implemented within the HE framework. For this purpose, the Multi-Layer Perceptron (MLP)-based logistic regression model was used as the plaintext model. We utilized two types of datasets employed in emotion recognition for the evaluation: the WESAD dataset [63] and the Hide-and-seek dataset collected in workplaces requiring emotional workload [58]. Regarding the second research question, we evaluated the model's practicality from two aspects. First, we demonstrated that the encrypted emotion recognition model could also be utilized for model personalization. In cases where physiological data or device usage history data reveal significant individual differences, the inference performance of the global model may decline. Therefore, the importance of model personalization in emotion recognition is emphasized, and we verified whether a model encrypted with HE could also be applied to personalization. We adopted a co-training approach to enhance model performance, enabling semi-supervised learning that allows the model to be used without collecting separate labels from new users. Furthermore, we empirically demonstrated that our HE-based model achieves a short inference time when deployed on a GPU server, making it suitable for real-time service.

We demonstrate that stress/emotion recognition can be performed even when utilizing a dataset encrypted with HE. For the WESAD dataset, when using plaintext data, the model achieved an accuracy of 0.966 and an F1-score of 0.976. When using encrypted data, the model's accuracy and F1-score were 0.967 and 0.971, respectively. For the Hide-and-seek dataset, when using plaintext data, the model achieved an accuracy of 0.868 and an F1-score of 0.899. When using encrypted data, the model's accuracy and F1-score were 0.868 and 0.899, respectively. This demonstrates that the EER framework can protect data privacy while minimizing model performance degradation. Additionally, we show that it is possible to personalize the model even in an encrypted environment by training with unlabeled encrypted data. We assume that 100 of the participant datasets are unlabeled data and the rest are labeled data for evaluation. As a result, for the WESAD dataset, the accuracy and F1 score of the encrypted personalized model are improved by 0.011 and 0.014, respectively, compared to the encrypted generalized model, reaching 0.978 and 0.985. Similarly, for the Hide-and-seek dataset, the accuracy and F1-score of the encrypted generalized model were 0.868 and 0.899, whereas the accuracy and F1-score of the encrypted personalized model were 0.878 and 0.905, demonstrating a performance improvement of 0.001 and 0.006 after personalization. Furthermore, we empirically demonstrated that our homomorphic encryption-based MLP model achieves inference times suitable for real-time service. As a result, the total inference time of our proposed EER framework was measured to be 556 ms, including both encryption and decryption, as well as communication between the user and the server. Specifically, the encrypted model inference performed on a GPU server took only 455 ms, confirming its viability for real-time applications.

In summary, our EER framework satisfies the following three key properties: 1) It guarantees IND-CPA security, a standard notion of cryptographic semantic security; 2) It achieves practical performance that is competitive with existing state-of-the-art models; and 3) It supports real-time inference under HE. Beyond its technical contributions, we note that we propose a not only secure but also trustworthy and easy-to-use methodology from a usable security perspective. For example, applying the EER framework does not require users to provide additional personal information or to accept any degradation in service quality (e.g., reduced model accuracy or loss of real-time availability). Therefore, the EER framework can be applied regardless of the user's technical literacy, imposing minimal cognitive or psychological burden on the user.

To guide the reader through our work, we provide a detailed explanation of the EER framework's operation and implementation. Lastly, we discuss the theoretical and technical contributions, along with mentioning

in-situ feasibility and limitations of the EER framework. We publicly release the code implemented for the EER framework[1].

2 BACKGROUND AND RELATED WORKS

2.1 Emotion Detection in Workplaces

In the workplace, sensing has been utilized for various purposes. For example, workers' stress or anxiety and their affect or mood are monitored for their mental and physical well-being. Such sensing-based monitoring has been applied to workers in various workplace environments. Booth et al. [7] analyzed movement patterns based on data collected through Fitbit, smartphone, and OMsignal garment-based sensors to monitor the stress of hospital workers. Data such as step count, heart rate, sleep, acceleration, fat burn, cadence, breathing, and sitting time is used to capture the workers' behavior consistency by generating time-series clusters of human bio-behavioral signals [25]. Moreover, sensing is also utilized to assess and forecast knowledge workers' stress, mood, health, and physiological status. Umematsu et al. [69] conducted research to forecast the stress and health of high-tech company workers by using data such as skin conductance, skin temperature, and acceleration. Furthermore, Dartmouth's research team developed a mobile application to collect data including audio data to infer conversations, Bluetooth data to monitor sociability, and GPS data for mobility tracking [53]. In addition to detection, sensing has also been used to measure cognitive load. To assess the cognitive load of workers related to interruption management, consumer wearables are utilized [62], as well as biometric data and computer interaction histories [84].

In the workplace, sensing is primarily used for purposes that benefit workers, such as improving their well-being or productivity. However, concerns about the potential drawbacks of sensing have also been continuously raised. For example, data collected through sensing, such as Photoplethysmography (PPG), Electrodermal Activity (EDA), skin temperature, physical activity, sleep, phone usage, and acceleration, provide information related to workers' physiological and health status. However, when this data is secondarily processed, it may reveal sensitive information about workers' health conditions or the presence of diseases. A worker's health status is not only private information in itself but also highly critical, as it is directly linked to employment. In fact, workplace sensing has been used to monitor job performance or to identify the roles of individual workers. Swain et al. researched to understand individual roles by collecting data on workers' location, sleep duration, and phone usage through multisensors, such as wearables, phones, and beacons [19]. Additionally, Mirjafari et al. verified that mobility, activity, phone usage, and physiological data collected through Garmin, phones, and Gimbal Beacons can differentiate between high and low job performers [51]. Thus, even if sensing is employed for worker-centric purposes, it is essential to obtain consent from workers before collecting any data and to strictly limit the scope of data usage. Nonetheless, some concerns remain, as the implementation of these practices still relies to a certain extent on the employer's ethical considerations.

2.2 Privacy-preserving Emotion Detection

As the importance of emotional privacy has been emphasized [61], various methods have been employed to address the privacy issue. Among these, federated learning is one of the most actively utilized approaches. Federated learning protects personal information by localizing raw data on devices and transmitting only gradients and weights to the global model. Chhikara et al. [17] proposed a decentralized framework using federated learning to monitor workers' mental health in the workplace by leveraging facial expressions and speech signals without compromising privacy. Similarly, Gahlan et al. [28] utilized federated learning to develop an emotion recognition model using privacy-sensitive physiological multimodal datasets involving EEG, GSR, ECG, and RESP, ensuring the security of personal information. Federated learning has also been widely applied in collecting sensitive user

[1]https://github.com/thrudgelmir/EER

data on mobile devices, authentication, and IoT applications [12, 46, 66, 77]. However, this approach presents two critical issues. First, while federated learning can protect users' personal data during the training phase by keeping it on local devices, it does not inherently safeguard the privacy of user data during the inference phase. In other words, federated learning is not designed to protect the privacy of users who interact with the model after deployment. Therefore, additional security mechanisms are required to ensure data privacy during inference. Second, it has been shown that gradient information sent from users to the server can potentially be exploited to infer meaningful information about the underlying data. This raises concerns especially in personalized learning scenarios, where gradient updates may inadvertently reveal sensitive personal attributes of individual users. As a result, federated learning alone is insufficient to address the core privacy challenges in emotion recognition services.

Differential privacy and adversarial learning have also been used to make the data difficult to identify. Differential privacy involves adding noise to the data to make it difficult to determine whether a specific user's data is included. It enables the implementation of privacy-preserving applications by incorporating random noise into wearable sensor data [38]. Similarly, adversarial learning anonymizes data using a generator and discriminator. Jaiswal et al. [34] demonstrated that demographic information could leak during emotion recognition tasks using mobile data and proposed an adversarial learning paradigm to prevent the model from learning private information. This approach has also been applied to speech emotion recognition. Testa et al. [67] proposed a model that creates additive noise for speech, maintaining transcription performance while preventing unintended surveillance. Additionally, adversarial learning has been used in image-based emotion recognition to prevent user identity from being identified by removing user-dependent information from the convolutional kernel of convolutional neural networks [54]. However, these techniques are likewise focused on preventing the leakage of personal data during the training process, and are not applicable for protecting the data of users during real-time service interactions.

Additionally, various methods have been used to protect the private information of user data collected in mobile environments. First, on-device deep learning has been employed to personalize models while preventing the transmission of sensitive personal information to global models [26, 39, 75]. In addition, although not specifically for emotion recognition tasks, there have been attempts in the HCI field to utilize HE for data encryption. Zhou et al. [83] proposed a method that uses HE to encrypt participant data with the secret key of a Crypt-Service Provider (CSP) to prevent the exposure of a user's location data. Our approach differs in that we propose encrypting data directly with the user's secret key, fundamentally blocking any third party other than the user and server from accessing the data.

2.3 Homomorphic encryption

HE is an advanced encryption scheme that allows arbitrary arithmetic operations to be performed directly on encrypted data. Specifically, homomorphic encryption is designed to satisfy the following property: for an arbitrary arithmetic circuit C and input messages $\{m_i\}_{i\in[1,t]}$, the encryption scheme ensures that

$$\mathsf{Dec}(\mathsf{Eval}(C, \{\mathsf{Enc}(m_i)\}_{i\in[1,t]})) = C(\{m_i\}_{i\in[1,t]}),$$

where Enc and Dec denote the encryption and decryption algorithms, respectively, and Eval is an evaluation algorithm that takes an arithmetic circuit and encrypted messages as input and returns an encrypted result. Since any arithmetic circuit can be composed of additions and multiplications, homomorphic encryption can be characterized as a cryptographic scheme that supports unbounded additions and multiplications directly over encrypted data.

As a cryptographic primitive, homomorphic encryption must also satisfy standard notions of data security. Specifically, any party that does not possess the secret key should be unable to infer any partial information about the underlying data from the ciphertext. This level of security is formally captured by the notion of IND-CPA

(indistinguishability under chosen-plaintext attack) security. The fact that a homomorphic encryption scheme satisfies IND-CPA security implies that even a server performing computations on ciphertexts—without access to the data owner's secret key—cannot learn any information about the encrypted data, while still being able to carry out arbitrary arithmetic computations over it. Using these properties of HE to infer and train ML models without exposing the user's data is known as privacy-preserving machine learning (PPML). A PPML scenario works as follows: (1) the user encrypts the data with a public key and sends the ciphertext to the cloud server, (2) the cloud server uses the HE to infer/train on the encrypted data and sends the encrypted results back to the user, and (3) the user decrypts the ciphertext using the decryption key to obtain the results. This means that the user does not need to do any additional computation beyond encrypting/decrypting the data, and all computation is done in the cloud as a ciphertext. Moreover, the server does not learn any partial information about the users throughout this process.

Following Gentry's groundbreaking research, numerous HE schemes have been developed [9–11, 13, 14, 18, 21]. Among these, the CKKS (Cheon-Kim-Kim-Song) scheme is particularly well-suited for handling real-number data. It is constructed based on the hardness of the Ring variant of the Learning with Error (Ring-LWE) problem, which is well-known as a computationally hard lattice-based problem. Since real values are predominantly used in artificial intelligence computations, CKKS has become a widely adopted HE algorithm in the field of PPML, which focuses on performing AI computations while ensuring data privacy. CKKS scheme encrypts a vector of length n whose elements are complex numbers. n is a parameter related to the security and efficiency of the encryption scheme, and $n = 2^{15}, 2^{16}$ is standard. CKKS supports the following four basic operations for $v \in \mathbb{C}^n$:*addition, plaintext multiplication, ciphertext multiplication* and *rotation*. For vectors $v, w \in \mathbb{C}^n$, $v + w$, $v \cdot w$ represent the elementwise addition, multiplication. Additionally, $\rho(v, r)$ denotes the left cyclic shift of v by r steps, defined as $(v_r, v_{r+1}, \cdots, v_{n-1}, v_0, \cdots, v_{r-1})$. If $\mathsf{ct}_1, \mathsf{ct}_2$ denotes the ciphertexts of v, w, respectively, the functionality of each operation is as follows:

- $\mathsf{Add}(\mathsf{ct}_1, \mathsf{ct}_2) = \mathsf{ct}_{\mathsf{add}}$: $\mathsf{ct}_{\mathsf{add}}$ is decrypted to $v_1 + v_2$. It can be abbreviated to $\mathsf{ct}_1 + \mathsf{ct}_2$.
- $\mathsf{PMult}(\mathsf{ct}_1, v_2) = \mathsf{ct}_{\mathsf{pmult}}$: $\mathsf{ct}_{\mathsf{pmult}}$ is decrypted to $v_1 \cdot v_2$. It can be abbreviated to $v_2 \cdot \mathsf{ct}_1$.
- $\mathsf{CMult}(\mathsf{ct}_1, \mathsf{ct}_2, \mathsf{mk}) = \mathsf{ct}_{\mathsf{cmult}}$: $\mathsf{ct}_{\mathsf{cmult}}$ is decrypted to $v_1 \cdot v_2$. It can be abbreviated to $\mathsf{ct}_1 \cdot \mathsf{ct}_2$.
- $\mathsf{Rot}(\mathsf{ct}_1, r, \mathsf{rk}_r) = \mathsf{ct}_{\mathsf{rot}}$: $\mathsf{ct}_{\mathsf{rot}}$ is decrypted to $\rho(v_1, r)$.

Among these operations, CMult and Rot require *key-switching* process. The ciphertext generated by multiplication and rotation operations requires a different decryption key than the original decryption key. Therefore, the server should convert the changed decryption key to the original decryption key. This process is called *key-switching*, which is costly and requires the server to pre-calculate the keys required by the model and receive them in advance from the user. We define the switching key required for the multiplication operation as mk and the switching key required for the r-step rotation operation as rk_r. The detailed mathematical structure of ciphertexts and the specific homomorphic encryption algorithms are not essential for understanding the core contributions of this paper and are therefore omitted. Interested readers are referred to the following works for further details.

3 PRIVACY-PRESERVING EMOTION DETECTION MODEL

3.1 Design Objectives and Proposed Solutions

We demonstrate the feasibility of providing privacy-preserving real-time emotion recognition services that protect the user's data from the server while maintaining performance comparable to state-of-the-art models on multimodal emotion recognition tasks. Furthermore, we show that personalized modeling can be achieved within a reasonable computational time while preserving the privacy of user-specific data.

The purpose of this study is to address the following three key issues that are commonly identified as major obstacles in research and system development within the field of emotion/stress detection. Our proposed EER

framework addresses privacy concerns, facilitates the development of reliable personalized models, and reduces the workload for participants, making it a practical solution for EER systems.

3.1.1 Privacy of Worker/User Data. The potential violation of participants' privacy is a significant challenge in the field of data collection, particularly when involving sensor data and video or behavioral data. This concern arises from the sensitive nature of the data, which often includes personal or identifiable information. As a result, obtaining explicit consent from participants becomes not only a legal requirement but also an ethical necessity. However, the process of securing agreement can be complex, especially when dealing with large-scale or real-time data collection scenarios. Moreover, even if consent is obtained, users' concerns about the potential misuse of data have been pointed out as a significant issue. To address this issue, it is crucial to develop privacy-preserving methods that allow data collection without compromising the anonymity and security of the participants. By employing HE, sensitive sensor and behavioral data can be securely processed in its encrypted form, ensuring participants' privacy without exposing raw data, while also meeting legal and ethical requirements.

3.1.2 Necessity of Personalized Model. The substantial individual variability observed in these data is also a critical challenge in providing high-quality personalized services using emotion or stress data. The performance of machine learning models trained on generalized emotion/stress datasets often depends heavily on whether a specific individual's data is included in the training set. This issue highlights the limitation of generalized models in addressing the nuanced and unique patterns of individual behavior and emotional responses, resulting in inconsistent and suboptimal service quality. To address this challenge, it is essential to focus on the development of personalized models that leverage individual-specific data for training. Such models can capture the unique characteristics of each user, ensuring more stable and reliable performance. The framework enables the creation of personalized models by securely utilizing individual-specific data. HE ensures that data remains encrypted during computation, allowing models to learn unique patterns without risking data leakage.

3.1.3 Difficulty of Data Labeling. The burden placed on workers when they are required to perform labeling tasks, such as self-reporting, is another significant issue in data collection and model personalization. These tasks are often time-consuming and tedious, making them impractical for large-scale or continuous data collection. This burden not only discourages participation but also limits the availability of labeled data, which is crucial for training personalized models. To address this challenge, leveraging semi-supervised learning techniques, such as co-training, offers a promising solution. By implementing co-training within a framework using HE, it becomes possible to personalize models without requiring workers to provide explicitly labeled data. Instead, unlabeled data can be utilized effectively, reducing the workload for participants while still enabling meaningful model training. This approach ensures privacy protection through HE and minimizes the dependency on fully labeled datasets, thus striking a balance between personalization and practicality in real-world applications.

3.2 Encrypted Emotion Recognition (EER) Framework

This section describes the encrypted emotion recognition (EER) framework designed to protect user information during emotion recognition services. We define the roles of the server and the user, and distinguish between the generalized and personalized frameworks depending on whether personalization is applied. We detail the data flow and computational environment for each phase. The specific model architecture and implementation details are elaborated in the following section.

3.2.1 System and Security Model. In our system model, there are two parties: the server and the user. The server plays two primary roles: during the online phase, it provides emotion analysis services to the user (who is considered a worker), and during the offline phase, it performs model training for the emotion analysis service. For model training, the server can utilize publicly available datasets, which do not require encryption since they

are already in plaintext. The user, on the other hand, is assumed to lack the computational resources to perform the service-related computations independently. The user's role is limited to preprocessing their data, generating keys, and encrypting their data. Since the user is reluctant to share their multimodal data with the server in plaintext, the server cannot access any user data in plaintext during service provision or model personalization. Instead, the server can request encrypted data from the user either during the online service provision phase or during the offline model personalization phase. It is assumed that the publicly available datasets are labeled, with each data point annotated to indicate its corresponding emotional state. On the other hand, the data held by the user is assumed to be unlabeled, meaning that no annotations are provided for each data point. This assumption aligns well with real-world workplace environments, where maintaining data privacy while enabling effective service provision is a critical requirement.

It is important to note that the phase requiring real-time processing is the online phase, during which the user is actually being served. In contrast, the offline phase occurs when the user is not actively using the service and serves as a preparation stage for the online phase. Therefore, the model personalization process does not need to be performed in real-time, as it is part of the offline preparation for enabling real-time inference. Furthermore, considering that both the model personalization and inference processes are performed on encrypted data by the server, the user's computational burden remains minimal—as long as the user device can perform basic encryption and decryption operations. Thus, even if the user uses resource-constrained devices, we assume that it does not impose a significant computational load on the user.

3.2.2 Framework Overview. The framework is divided into two types based on the presence or absence of model personalization: the *generalized framework* and the *personalized framework*. In the generalized framework as in Fig. 1, the server trains a model for emotion analysis using publicly available datasets during the offline phase and prepares it for service provision. In the online phase, the server provides emotion analysis services by performing inference on the user's encrypted data. In the offline phase, the server does not request encrypted data from the user and trains the model on publicly available datasets in plaintext. As a result, there is no need for encryption or homomorphic operations during training, and the resulting model weights are also in plaintext. In the online phase, the user encrypts multimodal data for emotion analysis in real time and sends the encrypted data to the server. The server then performs inference using the plaintext model on the encrypted data, producing encrypted inference results. The server has no access to any intermediate or final results during the computation. The encrypted inference result is sent back to the user, who decrypts it using their private key to obtain the final outcome. Importantly, only the user can access the inference result, ensuring complete privacy throughout the process.

In the personalized framework as in Fig. 2, an additional personalization step is performed during the offline phase to provide services tailored to each individual user. Initially, as in the generalized framework, the server trains an emotion analysis model using publicly available labeled datasets. After this step, the server receives encrypted multimodal data from the user and uses it for personalized training. Since the user's data remains encrypted throughout this process, the resulting personalized model weights are also encrypted using HE. In the online phase, the workflow is similar to that of the generalized framework. The user encrypts the data they wish to analyze and sends it to the server. The server then performs inference using the encrypted personalized model. The inference result, still encrypted, is sent back to the user, who decrypts it using their private key to obtain the final result. This approach ensures that both the user's data and the personalized model remain secure and private throughout the entire process.

The personalized framework introduces two major differences from the generalized framework from an HE-based implementation perspective:

- Encrypted personalized training: During the offline phase, personalized training must be performed using homomorphic operations on encrypted data.

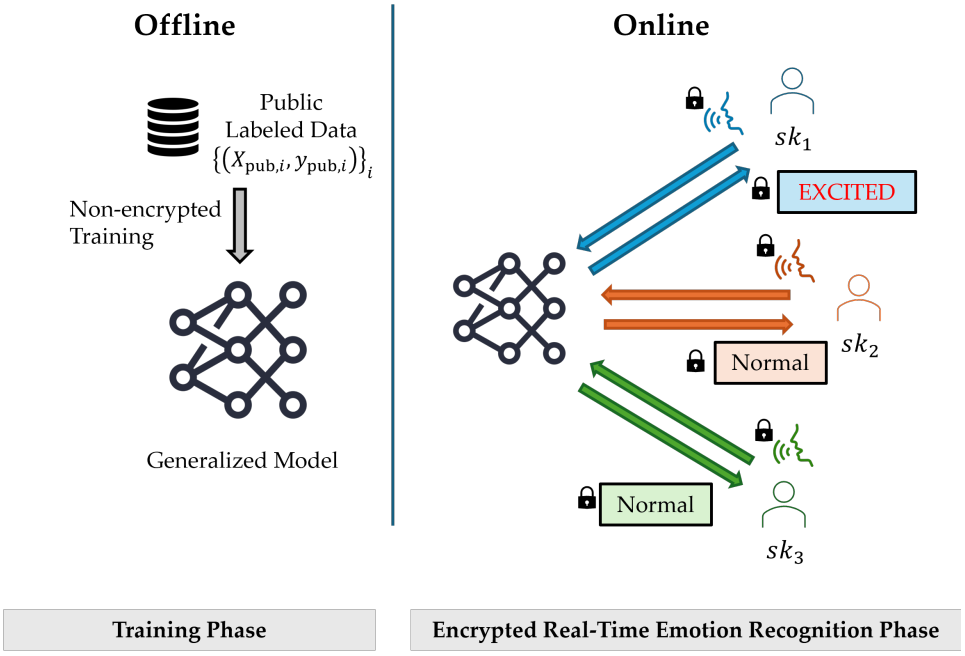


Fig. 1. Generalized framework of proposed emotion/stress detection system

- Encrypted inference with encrypted model weight: Since the resulting model weights remain encrypted after the personalized training phase, both the model weights and the input data must remain encrypted during the online inference phase.

The key challenge here is designing an efficient and high-performing personalized training method under HE. To address this, we apply a *co-training* technique. The co-training technique is a semi-supervised learning method in which the features of the data are divided into two distinct views, and two independent models are trained separately for each view. During the learning process, each model incorporates the inference results of the other, enabling collaborative learning and improving overall performance. Notably, co-training under HE has not been explored in prior research, making this a technically significant contribution of our work. There are many variations of the co-training technique, and we choose the *cross-view labeling* technique for two views in our proposed framework.

3.3 Technical Implementations of EER Model

This section discusses the implementation details of the EER system, with a particular focus on how it is implemented using CKKS HE. From a technical perspective, we emphasize two key points. First, we propose a lightweight model that mitigates the substantial computational overhead of homomorphic encryption, achieving real-time service capability while maintaining performance comparable to state-of-the-art methods. Second, we present a concrete homomorphic encryption-based algorithm that enables the co-training procedure to be executed directly on encrypted data, thereby supporting personalization without compromising privacy. To the best of our knowledge, this is the first work to propose a co-training algorithm that operates over homomorphically encrypted data.

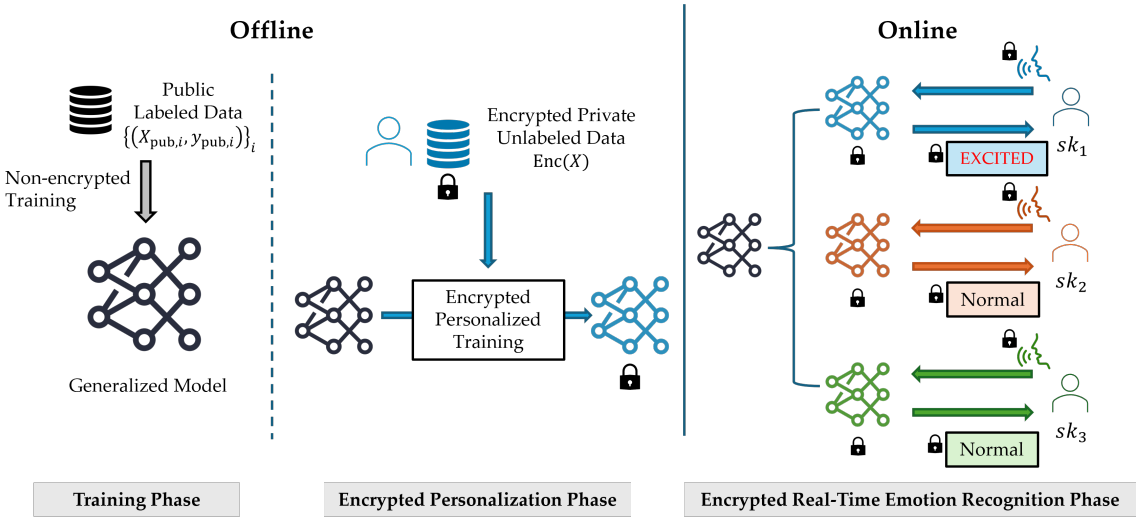


Fig. 2. Personalized framework of proposed emotion/stress detection system

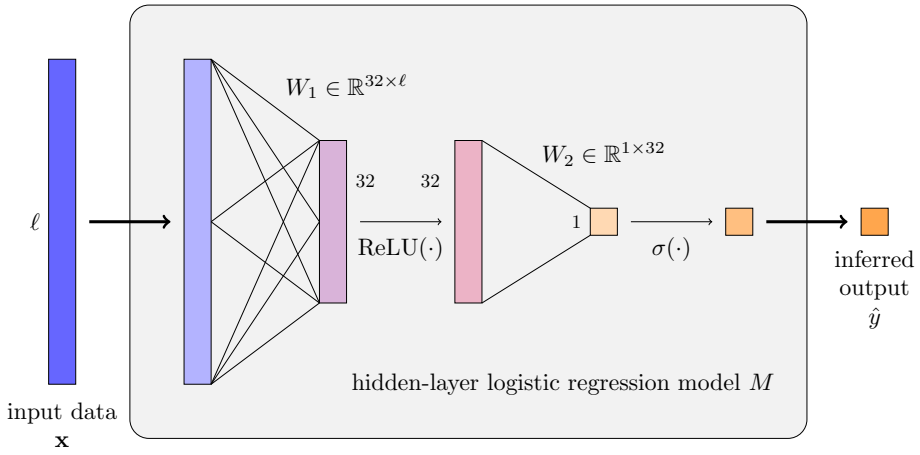


Fig. 3. MLP-based logistic regression model in EER framework

3.3.1 MLP-based Logistic Regression Model. We construct an emotion detection model suitable for computation with HE-based on the following three criteria.

- The model possesses an oblivious property, where both training and inference computations remain unaffected by the data.
- The model's performance is comparable to the best results achieved in the emotion detection task.
- The computational overhead when operating on HE must be minimized as much as possible.

Our model is a *MLP-based logistic regression*, which is a logistic regression model with a single MLP layer using the ReLU activation function, which is depicted in Fig. 3. Notably, this model does not perform computations that

depend on the input data during execution, meaning it inherently possesses the oblivious property. Additionally, since the model consists of only two layers, it has minimal computational overhead. Due to its inherent obliviousness, executing it under HE does not introduce additional computational costs, making it highly efficient. Furthermore, as demonstrated in Section 5, this model achieves performance comparable to state-of-the-art models in the emotion detection task. Given these advantages, we design our EER model around this architecture.

3.3.2 Generalized Framework. In the generalized framework, since training is conducted on public data, it follows the same approach as conventional AI systems and is performed in plaintext. Therefore, the model is trained by applying the gradient descent algorithm to update the weights of the MLP-based logistic regression model shown in Fig. 3. The details of this training process are omitted, and we focus solely on how inference is performed on encrypted data during the online phase. We restrict to the top ℓ features based on the ANOVA F-value to account for the limited capacity of HE. The specific ℓ values for each dataset will be shown in Section 5. We assume that inference is conducted simultaneously on $d = 128$ data points, each with ℓ features.

For the d data points held by the user, denoted as $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{d-1} \in \mathbb{R}^{\ell}$, we construct a matrix $X = [X_{i,j}] = [\mathbf{x}_0|\mathbf{x}_1|\cdots|\mathbf{x}_{d-1}] \in \mathbb{R}^{\ell \times d}$. To efficiently encode this data into a ciphertext, we use row-major packing, where the rows of X are concatenated sequentially into a one-dimensional vector $\mathbf{v} \in \mathbb{R}^{\ell d}$. Specifically, if we denote the rows of X as X_i, then $\mathbf{v}$ is defined as $\mathbf{v} = (X_0|X_1|\cdots|X_{\ell-1}) \in \mathbb{R}^{\ell d}$. The user then encrypts $\mathbf{v}$ to generate a ciphertext: $\mathsf{ct} = \mathsf{Enc}(\mathbf{v})$ and transmits it to the server for inference.

The server should perform the following computation:

$$\hat{\mathbf{y}} = \sigma\left(W_2 \cdot \mathrm{ReLU}(W_1 \cdot X)\right) \in \mathbb{R}^{1 \times d}$$

To achieve this, the process is divided into four layers: first fully connected (FC) layer $W_1 \in \mathbb{R}^{32 \times \ell}$, ReLU activation function, second fully connected (FC) layer $W_2 \in \mathbb{R}^{1 \times 32}$, and sigmoid function. Each layer is executed as follows:

(1) First FC layer (W_1): FC layer is performed using the rectangular matrix multiplication algorithm introduced in [36]. Specifically, [36] outlines an algorithm for the matrix multiplication of a rectangular matrix A of size $k \times d$ with a square matrix B of size $d \times d$, where k is a divisor of d and satisfies $k < d$. The algorithm involves applying a permutation linear transformation to each matrix, with the transformation process detailed in Algorithm 1. The complete matrix multiplication procedure is described in Algorithm 2, and the resulting output matrix takes the form $[AB|AB|\cdots|AB]^t \in \mathbb{R}^{d \times d}$.
(2) ReLU activation function ($\mathsf{ReLU}(\cdot)$): The sign function $\mathsf{sign}(x)$ is approximated using a composition of the three small minimax polynomials via the composite function method introduced in [41]. Using the formula $\mathsf{ReLU}(x) = x \cdot \mathsf{sign}(x)$, The $\mathsf{ReLU}(\cdot)$ function is computed using three polynomial evaluations followed by one multiplication.
(3) Second FC layer (W_2): The second fully connected (FC) layer can be computed using a simple inner product operation. In ciphertext, the inner product is computed using one PMult operation with the weight vectors, followed by a rotation-and-sum operation requiring $\lceil \log t \rceil$ Rot operations, where t is the length of the vector involved in the inner product. For our task, as the inner product involves two vectors of dimension 32, a total of $\log(32) = 5$ rotations are required.
(4) sigmoid function ($\sigma(\cdot)$): The sigmoid function is approximated as a polynomial using the Chebyshev approximation method. The approximation range is determined based on the range observed immediately prior to the sigmoidal activation step during inference on public data, and the polynomial degree is selected to ensure an approximation error of less than 10^{-5} within this range.

3.3.3 Personalized Framework. In the personalized framework as in Fig. 4, features must be divided into two clusters to enable the co-training process. The server applies the K-means clustering algorithm to partition the

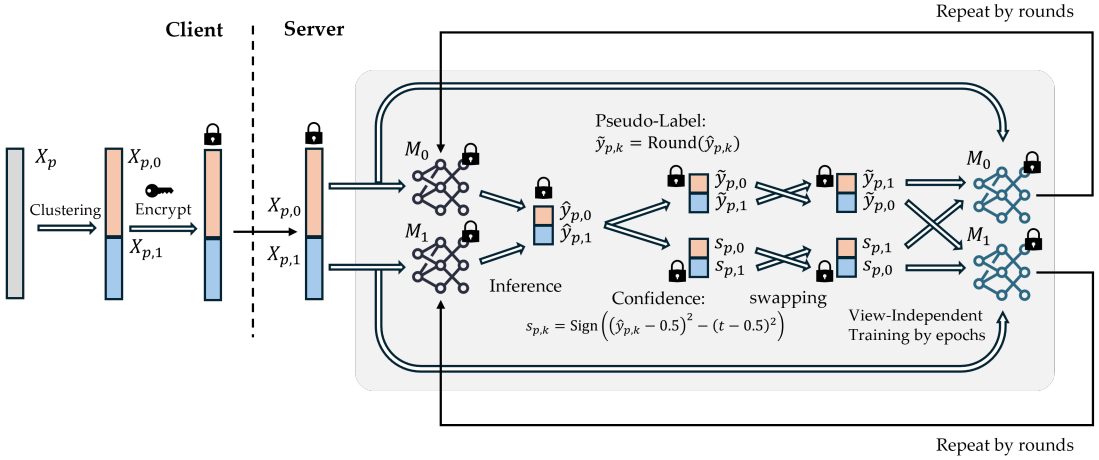


Fig. 4. Homomorphic co-training process in personalized framework

feature set into two clusters for co-training based on the public dataset. The server utilizes the K-means clustering algorithm to create two feature clusters based on $\mathbf{x}_p$ of features for co-training, after which the server informs the user about the features corresponding to the two views. When the number of features in each view is ℓ_0, ℓ_1, each data can be divided into $\mathbf{x}_p \rightarrow (\mathbf{x}_{p,0}, \mathbf{x}_{p,1}) \in \mathbb{R}^{\ell_0} \times \mathbb{R}^{\ell_1}$. The server then constructs two independent MLP-based logistic regression models, each corresponding to one of the two views. Using the features from the public dataset associated with each view, the server independently trains these models, resulting in two non-personalized generalized models.

To handle the data of each view independently while still packing them into a single ciphertext, we employ the *parallel packing method*. This method encodes two vectors, $\mathbf{v}_0 = (v_{0,i})_{i\in[0,n/2)} \in \mathbb{R}^{n/2}$ and $\mathbf{v}_1 = (v_{1,i})_{i\in[0,n/2)} \in \mathbb{R}^{n/2}$, into a single vector $\mathbf{w} = (w_i)_{i\in[0,n)} \in \mathbb{R}^n$. Specifically, the even-indexed positions store elements from $\mathbf{v}_0$, while the odd-indexed positions store elements from $\mathbf{v}_1$. This encoding is formally expressed as $w_{2i} = v_{0,i}, w_{2i+1} = v_{1,i}$. For convenience, we denote this parallel packing operation as $\mathbf{w} = (\mathbf{v}_0/\mathbf{v}_1)$. If a homomorphic circuit C is to be applied separately to $\mathbf{v}_0$ and $\mathbf{v}_1$, it can instead be executed on the packed vector $\mathbf{w} = (\mathbf{v}_0/\mathbf{v}_1)$ with the following adjustment: all rotation steps in the original circuit are replaced with twice the original rotation step size, while all other operations remain unchanged. Let $\tilde{C}$ denote this modified circuit. Moving forward, we consider applying C to $\mathbf{v}_0$ and $\mathbf{v}_1$ equivalent to applying $\tilde{C}$ to $\mathbf{w}$ in our computations.

The process of encrypting the user's data required for model personalization is as follows. Similar to the generalized framework, we pack $d = 128$ data points into a single ciphertext. However, to handle the two views independently, we consider the following matrices: $X_0 = [\mathbf{x}_{0,0}|\mathbf{x}_{1,0}|\cdots|\mathbf{x}_{d-1,0}], X_1 = [\mathbf{x}_{0,1}|\mathbf{x}_{1,1}|\cdots|\mathbf{x}_{d-1,1}]$. Each of these matrices is encoded using row-major packing, resulting in two vectors, $\mathbf{v}_0 \in \mathbb{R}^{\ell_0}$ and $\mathbf{v}_1 \in \mathbb{R}^{\ell_1}$. These vectors are then parallel packed into a single vector: $\mathbf{w} = (\mathbf{v}_0/\mathbf{v}_1)$.. (To ensure proper alignment, any unused slots are zero-padded.) The encrypted ciphertext is then generated as: $\text{ct} = \text{Enc}(\mathbf{w})$, which is subsequently transmitted to the server for further processing.

The server performs co-training on the encrypted data, iterating through multiple rounds to complete the personalization process. Each round consists of the following three steps: pseudo-labeling, cross-view filtering, and view-independent training. The following paragraphs provide a detailed explanation of each step.

Pseudo-Labeling. In this step, the encrypted unlabeled training data for personalization is inferred using the model from the previous round, and the results are rounded to generate *pseudo-labels*. To this end, inference is first performed on the current model to obtain the inferred result. This process is similar to the forward process in the generalized framework, where inference is conducted on encrypted data using the MLP-based logistic regression model. However, a key difference is that in the personalized framework, the model weights are also encrypted. The weight matrices, $W_{1,0} \in \mathbb{R}^{32\times\ell_0}$, $W_{1,1} \in \mathbb{R}^{32\times\ell_1}$, $W_{2,0}$, and $W_{2,1} \in \mathbb{R}^{1\times 32}$, are all packed using the row-major packing method. Matrix multiplications involving these weights can be efficiently computed using the method proposed by [36]. After performing these computations, we obtain the inferred results for each view $\hat{\mathbf{y}}_0, \hat{\mathbf{y}}_1$. To generate pseudo-labels, we apply a sign function to threshold the values: $\tilde{\mathbf{y}}_i = \mathsf{sign}(\hat{\mathbf{y}}_i - \frac{1}{2})$. If the inferred value is greater than 0.5, the pseudo-label is set to 1, and if it is less than 0.5, it is set to 0. These pseudo-labels serve as the labels for the encrypted unlabeled data, which will be used in the view-independent training process for this round. All these processes are performed while maintaining the parallel packing format.

Cross-View Filtering. In this step, based on the reliability of the inferred results obtained from the two independent models, each data point is filtered to determine whether it should be added to the training dataset of the opposing model, which has a different view. To this end, the confidence of the pseudo-labels is computed, followed by a swapping operation that exchanges pseudo-labels and confidence values between the two models. This swapping process ensures that each model incorporates pseudo-labels generated by the other model, thereby reinforcing co-training. The confidence metric determines whether a pseudo-label produced by one view's model is sufficiently accurate to improve the training performance of the other view's model. This confidence score plays a crucial role in view-independent training, as it decides whether a given pseudo-label and its corresponding data should be included in the training process. This mechanism is fundamental to co-training, as it enables knowledge transfer between the two models, allowing each to enhance the other's performance. The whole process is depicted in Fig. 5.

Before starting the co-training process, the server determines and fixes a confidence parameter $0 < t < 0.5$. The confidence of each pseudo-label is then evaluated based on whether the absolute difference between $\hat{\mathbf{y}}_i$ (the inferred value) and its corresponding pseudo-label is smaller than t. Computing this confidence is equivalent to checking whether each $y_{p,i}$ value falls within the interval $[t, 1-t]$. If a value lies within this range, the output is 0; otherwise, the output is 1. This is computed using the following equation:

$$\mathsf{Enc}(\mathbf{s}_i) = \mathsf{sign}\left(\left(\mathsf{Enc}(\hat{\mathbf{y}}) - \frac{1}{2}\right)^2 - \left(t - \frac{1}{2}\right)^2\right)$$

In the view-independent training phase, only the data points where $s_{p,i} = 1$ will be used for training.

After this step, a swapping operation is performed to exchange the positions of the pseudo-labels $\tilde{\mathbf{y}}_0$ and $\tilde{\mathbf{y}}_1$, as well as the positions of the confidence values $\mathbf{s}_0$ and $\mathbf{s}_1$. Since all previous operations have been conducted under parallel packing, the pseudo-labels and confidence values are packed as: $\tilde{\mathbf{y}} = (\tilde{\mathbf{y}}_0/\tilde{\mathbf{y}}_1), \mathbf{s} = (\mathbf{s}_0/\mathbf{s}_1)$. To perform the swapping operation, we need to construct the swapped versions: $\tilde{\mathbf{y}}' = (\tilde{\mathbf{y}}_1/\tilde{\mathbf{y}}_0), \mathbf{s}' = (\mathbf{s}_1/\mathbf{s}_0)$. Since swapping requires interchanging values between even-indexed and odd-indexed positions, this can be efficiently achieved using the following operation: $\mathbf{m} = (\mathbf{1}_{n/2}/\mathbf{0}_{n/2})$. The corresponding computation is then performed as follows, which is also depicted in Fig. 6:

$$\tilde{\mathbf{y}}' = \mathsf{Rot}(\mathbf{m}\cdot\mathsf{Enc}(\tilde{\mathbf{y}}), -1) + \mathbf{m}\cdot\mathsf{Rot}(\mathsf{Enc}(\tilde{\mathbf{y}}), 1), \quad \mathbf{s}' = \mathsf{Rot}(\mathbf{m}\cdot\mathsf{Enc}(\mathbf{s}), -1) + \mathbf{m}\cdot\mathsf{Rot}(\mathsf{Enc}(\mathbf{s}), 1)$$

View-Independent Training. In this step, each view's model is independently trained using gradient descent with the dataset with pseudo-label selected in the previous step. The core of our training algorithm is to eliminate the influence of untrusted data on the weight updates. We address this issue by multiplying the derivative of the loss, dL/dz, with a selection value. Specifically, the loss function uses the binary cross entropy function $H(y, \hat{y})$,

the output activation is sigmoid $\sigma(x)$, and the MLP activation is $\mathsf{ReLU}(x)$. The gradient descent formula for each weight in this case is as follows.

$$L(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N} \sum_{i=1}^{N} H(y_i, \hat{y}_i),$$

$$\frac{\partial L}{\partial W_2} = \frac{1}{N} \cdot (\hat{\mathbf{y}} - \mathbf{y}) \cdot A_1^T, \quad \frac{\partial L}{\partial W_1} = \frac{1}{N} \cdot (W_2^T \cdot (\hat{\mathbf{y}} - \mathbf{y})) \odot \mathsf{Sign}(Z_1) \cdot X^T,$$

where $A_1 = \mathsf{ReLU}(W_1 \cdot X) = [\mathbf{a}_{1,0} | \mathbf{a}_{1,1} | \cdots | \mathbf{a}_{1,N-1}]$, $Z_1 = W_1 \cdot X$, $\mathbf{y} = (y_0, y_1, \cdots, y_{N-1})$, $\hat{\mathbf{y}} = (\hat{y}_0, \hat{y}_1, \cdots, \hat{y}_{N-1})$, X are input data, N is the number of input data, which is actually $N = d$. In the forward procedure (Algorithm 4), $A_1, \mathsf{Sign}(Z_1)$ has already been computed. For convenience, we denote $(\mathbf{y} - \hat{\mathbf{y}})$ as $\frac{\partial L}{\partial z}$. Since $\frac{\partial L}{\partial z}$ appears as a common factor in each gradient formula, it can be precomputed by multiplying $\frac{\partial L}{\partial z}$ with the selection value $\mathbf{s}$ prior to the gradient computation. This approach ensures that the untrusted term $(y_i - \hat{y})$ is effectively set to zero, thereby completely eliminate the impact of untrusted data.

In the case of independent training for each example, reducing the time spent in the backward procedure results in a significant reduction in the overall training time. To achieve this, the algorithm is slightly modified to optimize the time used during training. In rectangular matrix multiplication, the output matrix corresponds to a matrix in which the original matrix is replicated N/k times. This property is leveraged to improve efficiency. The forward pass output $\hat{\mathbf{y}} = \sigma(W_2 \cdot A_1)$, where $W_2 \in \mathbb{R}^{1 \times 32}$, is encrypted such that the actual encrypted result is $\hat{\mathbf{y}}$ repeated N times, represented as $[\hat{\mathbf{y}} \mid \hat{\mathbf{y}} \mid \cdots \mid \hat{\mathbf{y}}]^T \in \mathbb{R}^{N \times N}$. Therefore, instead of computing the gradient $\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial z} \cdot A_1^T$ using matrix multiplication, we calculate it element-wise as follows:

$$\frac{\partial L}{\partial W_2} = \sum_{i=1}^{N} (\hat{y}_i - y_i) \cdot \mathbf{a}_{1,i} = \sum_{i=1}^{N} \mathbf{a'}_{1,i}$$

, where $\mathbf{a'}_{1,i} = (\hat{y}_i - y_i) \cdot \mathbf{a}_{1,i}$.

Each term $(\hat{y}_i - y_i) \cdot \mathbf{a}_{1,i}$ is stored in the matrix $\frac{\partial L}{\partial z} \odot A_1$, represented as $[\mathbf{a'}_{1,0} \mid \mathbf{a'}_{1,1} \mid \cdots \mid \mathbf{a'}_{1,N-1}]$,. The gradient $\frac{\partial L}{\partial W_2}$ can then be computed efficiently by transposing $\frac{\partial L}{\partial z} \odot A_1$, summing over its N rows, and aggregating the results. Next, the computation of $\frac{\partial L}{\partial W_1}$, specifically $W_2^T \cdot \frac{\partial L}{\partial z}$, can be performed using a single homomorphic multiplication. By defining $W_2' = [W_2 \mid W_2 \mid \cdots \mid W_2] \in \mathbb{R}^{N \times N}$, the following formula holds: $\frac{1}{N} \cdot W_2' \odot [\hat{\mathbf{y}} \mid \hat{\mathbf{y}} \mid \cdots \mid \hat{\mathbf{y}}]^T = W_2 \cdot \frac{\partial L}{\partial z}$. This formula enables efficient computation by leveraging the structure of W_2' and the repeated patterns in $\hat{\mathbf{y}}$. Also, as one popular method, X remains invariant throughout training, so precomputing some of the matrix multiplications involving X and its transpose X^T can reduce the computational cost of the forward and backward procedures. The transpose of a matrix can be performed using Algorithm 1 with the transpose transformation matrix as input. A detailed definition of the transpose matrix is introduced in [36]. The foward, backward, and co-training procedures are introduced in Algorithms 4, 5, and 7, respectively.

Finally, although not explicitly stated, our actual implementation incorporates bias terms within the weight matrices. To correctly execute both the forward and backward procedures when bias is included, we apply a masking operation that assigns a value of 1 to the indices requiring bias and 0 otherwise, ensuring the exact computation of the procedure.The augmented matrix $(X, \mathbf{1})$ can be efficiently represented within a ciphertext by incorporating a masking matrix M_b, where M_b is defined as follows: $M_b(i, j) = 1$, where $i = \ell$ and $0 \leq j < N$, $M_b(i, j) = 0$, otherwise. This ensures that when X has dimensions $\ell \times N$, the bias term is correctly embedded within the ciphertext representation. If it is necessary to remove the bias component, a masking matrix can be defined in a similar manner and applied homomorphic multiplication within the ciphertext to eliminate the bias term.

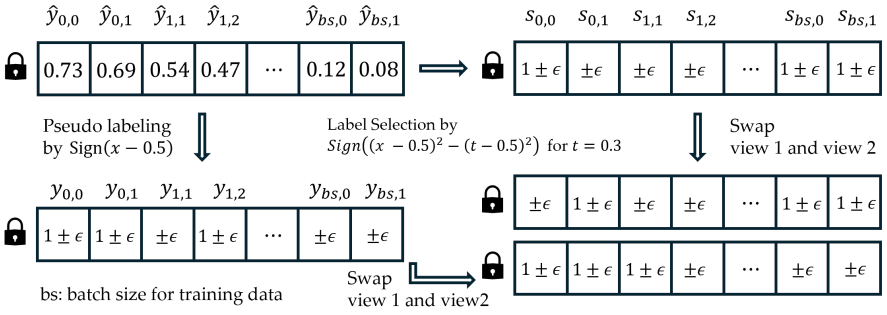


Fig. 5. Diagram for cross-view filtering process

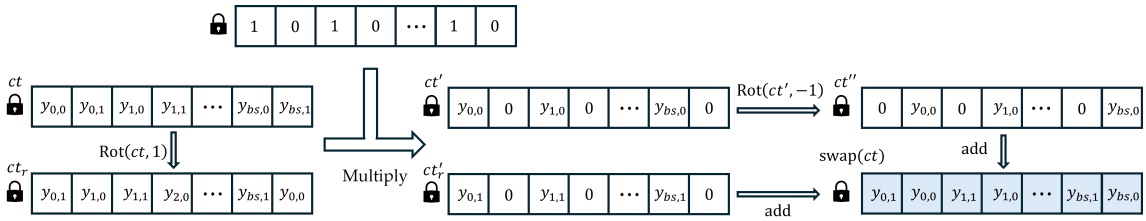


Fig. 6. Diagram for swapping process

4 MODEL EVALUATION

4.1 Dataset Description

We utilized two types of datasets to evaluate the performance of the EER framework. First, we used the WESAD dataset [63], which has been widely employed for stress and emotion recognition. Second, we used the Hide-and-seek dataset [58], collected in a workplace environment where data security issues are more sensitive. In this section, we provide detailed information about the datasets we used.

4.1.1 WESAD. The dataset was collected for stress and affect detection. 15 subjects participated in the data collection, which was conducted as an in-lab study. The collected dataset contains multimodal data including physiological (blood volume pulse, electrocardiogram, electrodermal activity, electromyogram, respiration, body temperature) and motion data (three-axis acceleration) collected by wrist- and chest-worn devices. The data collected from the chest-worn device were all sampled at 700 Hz, while the signals collected from the wrist-worn device were sampled at BVP (64 Hz), EDA (4 Hz), TEMP (4 Hz), and ACC (32 Hz). In addition, the three labels (neutral, stress, and amusement) were collected through a self-report method. The study protocol was designed to elicit the three different affective states. During the baseline condition, the subjects were sitting or standing at a table, and neutral reading material was provided. The subjects were watching funny video clips during the amusement condition. For the stress condition, Trier Social Stress Test (TSST) was provided which consists of public speaking and mental arithmetic tasks. The authors presented benchmarks for the three-class classification problem using standard machine learning methods.

4.1.2 Hide-and-seek. The dataset was collected to evaluate workers' emotional workload in emotional labor scenarios through multimodal sensing. The data collection involved a call center simulation study, where participants were exposed to manipulated customer behaviors acted out by confederates. It was collected from a

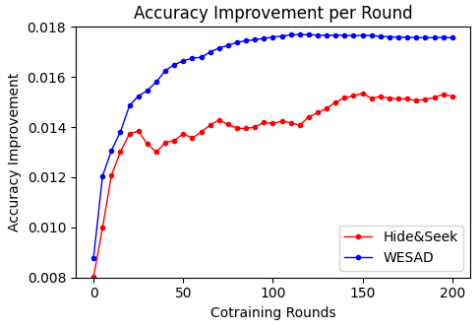


Fig. 7. Co-training accuracy improvements with plaintext

total of 31 workers who actually work in a call center and 4 customers. The worker participants wore wearable sensors capturing physiological signals (e.g., electrocardiogram, blood volume pulse, electrodermal activity, body temperature, interbeat interval, electroencephalogram) and motion data (e.g., three-axis acceleration), along with audio data including the conversation between a worker-participant and a customer-participant. In the experiment, the conditions were classified into neutral, shouting, and swearing conditions based on the role of the customer that the customer participant was instructed to perform. Emotional workload labels were assessed based on two methods: stimulus-based labeling (neutral, shouting, and swearing conditions) and the participants' self-reports. Benchmark results for both binary and three-class classification problems were reported using standard machine-learning techniques and well-known feature sets.

4.2 Model Validation Procedure

For both the WESAD and Hide-and-Seek datasets, we applied an early fusion strategy during preprocessing as performed in the previous studies [58, 63]. For example, audio signals of the Hide-and-Seek dataset were preprocessed and concatenated with physiological features to form a single unified feature vector. These fused multimodal inputs were then used to train both general and personalized models, without separate modality-specific processing. To validate the robustness of the models on unseen users, we used the Leave-One-Subject-Out (LOSO), which is the method to train the model on the data of all participants except one and test the model by using excluded data and repeat it for all participants.

5 RESULTS

In this section, we present numerical results of the proposed hidden-layer logistic regression architecture. Most experiments were performed in a GPU environment to achieve real-time inference; a detailed comparison and analysis with the CPU environment are provided in Section 5.3. GPU-based experiments were performed on an NVIDIA GeForce RTX 4090 (24 GB of memory) using HEaaN, a representative RNS-CKKS scheme library. For the CPU-based experiments, we used an AMD Ryzen 9 7900X 12-Core Processor (64 cores, 5.4 GHz) with 125 GB RAM and the Lattigo[1] library. Both systems operated on Ubuntu 20.04. Across these setups, we performed two main experiments on the Hide-and-Seek and WESAD datasets: LOSO Evaluation of the general model over ciphertext and LOSO evaluation of the co-training model over ciphertext.

5.1 Performance Comparison between Plaintext and Ciphertext Models

Figure 7 details the co-trained classification accuracy of the proposed model on plaintext on Hide-and-Seek and WESAD datasets under distinct training parameter configurations using 100 private unlabeled data. Each dataset

was trained with different settings for epoch, round, and confidence parameter. For the Hide and Seek dataset, we observe a 0.015 improvement in accuracy over the baseline as the number of rounds increases at 15 epochs. Similarly, for the WESAD dataset, we observe a 0.018 improvement in accuracy over the baseline as the number of rounds increases at 10 epochs.

Table 1 summarizes the classification accuracy, F1-scores, and latencies of the generalized model applied to plaintext and ciphertext data from the Hide-and-Seek and WESAD datasets, alongside the results of the personalized model. For the WESAD dataset, the ciphertext generalized model exhibits a classification accuracy and F1-score difference of only 0.001 and 0.005, respectively, compared to the plaintext model. Similarly, the personalized model achieves improvements of 0.011 in accuracy and 0.014 in F1-score over the ciphertext generalized model. For the Hide and Seek dataset, the ciphertext generalization model has almost no difference in accuracy and F1 score compared to the plaintext model, while the F1-score difference is 0.899. The personalized model shows an accuracy improvement of 0.01 and an F1-score difference 0.006 over the ciphertext generalized model. These results indicate that the proposed HE model suffers negligible performance loss due to encryption, while performance gains can be achieved, in the case of co-training.

Table 1. Encrypted co-training model accuracy, Improved accuracy compared to general models and training time. The hide-and-seek was trained for 25 rounds-15 epochs with a confidence variable t = 0.8. WESAD was trained for 10rounds-10epochs with confidence variable t=0.8.

Data	**Plain (Generalized Model)**			**Cipher (Generalized Model)**			**Cipher (Personalized Model)**			
	Accuracy	F1-score	Inference Time (ms)	Accuracy	F1-score	Inference Time (ms)	Accuracy	F1-score	Inference Time (ms)	Training Time (Cipher/Plain)
WESAD	0.966	0.976	0.151	0.967	0.971	333	0.978	0.985	455	329.12 s/49ms
Hide-and-Seek	0.868	0.899	0.207	0.868	0.899	333	0.880	0.908	455	1230.56 s/172ms

Comparing Figure 7 and Table 1, the training and inference process for encrypted data is very similar to that for plaintext. Given enough private data and training time, we can expect similar performance gains in encrypted settings.

By selectively conducting HE experiments on specific training parameter settings while running extensive plaintext experiments across various configurations, we further confirm that the training process itself remains unaffected by encryption. This highlights that HE can be seamlessly integrated into co-training without compromising accuracy or altering the learning dynamics.

For 128 data samples, the encrypted inference times for the WESAD and hide-and-seek datasets were recorded as 0.333 s and 0.455 s for the generalized and co-trained models, respectively. This difference is due to the fact that the weights in the generalized model are stored unencrypted, whereas in the co-trained model, the weights are encrypted. For the plaintext model, the inference time was measured to be 0.151 ms and 0.207 ms for the respective datasets, which is several thousand times faster than the ciphertext model. This discrepancy arises because each homomorphic encryption operation is inherently several orders of magnitude slower than its plaintext counterpart. These results highlight the necessity of adopting lightweight model architectures, such as our proposed MLP-based model, in order to ensure real-time performance under homomorphic encryption. Therefore, our framework, which adopts a lightweight model architecture and GPU implementation for real-time inference, achieves a latency of 0.455 seconds. This performance is sufficient to be considered real-time when compared to previous studies such as [33, 47], which reported latencies of 4 and 1.47 seconds, respectively.

For the WESAD dataset, it took about 329.12 s to train the co-trained model with parameters (10, 10, 0.8), while it took about 1230.56 s to train the co-trained model with parameters (25, 15, 0.8) for the hide-and-seek dataset. Specifically, each epoch took about 3.12 s on average. Personalized model training does not need to be performed

in real time, as it can be executed during the offline phase when the user is not actively interacting with the system. Since the required computation time is sufficiently short to be carried out during such idle periods, it does not pose a significant barrier to practical deployment.

5.2 Comparison with Baseline Models

We design a dedicated MLP-based model tailored to our task, rather than adopting existing emotion recognition models. To demonstrate its superior performance compared to other models, we conducted a performance-based comparison with two recent emotion recognition approaches: EmoNets [37] and the model proposed by Pogliaghi et al. [60]. All models were evaluated in plaintext environments.

Table 2. To ensure a fair performance comparison between our proposed MLP-based model and the baseline models, we applied the same feature extraction methods used in each baseline to our model. When comparing with EmoNets [37], we used the feature extraction method described in the WESAD dataset paper [63]. For the comparison with the model proposed by Pogliaghi et al. [60], we adopted the original feature extraction pipeline presented in their study.

	Feature Extraction in [63]		**Feature Extraction in [60]**	
	MLP (Ours)	EmoNets [37]	MLP (Ours)	Pogliaghi et al. [60]
Accuracy	0.966 ± 0.0014	0.707 ± 0.0085	0.924 ± 0.0017	0.921 ± 0.0051

Since EmoNets were originally developed under a different setting from our binary stress detection task, we adapted the model accordingly to enable a fair comparison based on our specific task. Sensor and speech data were preprocessed into timestep-level features, then integrated early into a single feature vector as input to a 1D CNN. The CNN outputs 2-dimensional softmax probabilities at each timestep. Ten consecutive outputs are concatenated to form a clip-level representation, which is then fed to an SVM for final classification. The clip-level label is generated by majority voting across the ten timestep labels. While this setup shares structural similarities with EmoNets in combining CNN and SVM, our model is based on early integration of modalities and single-stream feature processing, which makes late-fusion impractical in our setting—not due to external constraints but because of the inherent structure of our data pipeline. Due to the absence of external data streams and independent modality-specific processing in our setting, the late-fusion strategy used in EmoNets could not be effectively applied. Pogliaghi et al.'s model adopts a multi-task learning (MTL) framework using the same WESAD dataset as ours, and includes stress detection as one of its classification tasks. In the experiments presented in Section 5.1 and the comparison with the EMONET model, we used the feature extraction method proposed in the WESAD paper [63]. In contrast, Pogliaghi et al. employed their own feature extraction approach in their study. To ensure consistency, we used Pogliaghi et al.'s feature extraction method only when comparing our model with theirs, thereby enabling a direct comparison with their model under identical preprocessing conditions. The MTL model followed the original multi-task training paradigm, while our model was trained via standard supervised learning (excluding co-training), to focus solely on evaluating the performance of the MLP-based logistic regression structure.

Compared to our model, EmoNets exhibited noticeably lower performance. This is likely because the techniques employed in EmoNets were specifically optimized for their original task and environment, and did not transfer well to our binary stress detection setting. Specifically, due to the absence of external data streams and independent modality-specific processing in our setup, the late-fusion strategy employed in EmoNets could not be effectively applied. When compared to the model by Pogliaghi et al., our MLP-based model achieved comparable or slightly superior performance. These results indicate that our model is at least as effective as, if not better than, existing emotion recognition models in terms of classification performance. Furthermore, our MLP-based model has

a significantly simpler architecture than both EmoNets and the model by Pogliaghi et al., which gives it a distinct advantage in privacy-preserving settings. Its lightweight structure provides greater flexibility in adapting algorithms, such as co-training, for encrypted computation under homomorphic encryption. This makes our approach particularly well-suited for privacy-sensitive applications.

5.3 Comparison with CPU Environment and GPU Environment

This section analyzes the primary constraint of our framework—the necessity of a GPU environment for real-time inference. As shown in Table 3, this limitation is evident from the performance gap between CPU and GPU environments, which originates from the substantial computational overhead introduced by HE.

When evaluated on a GPU using existing datasets, the inference time increased from 0.207 ms (plaintext) to 333 ms for the generalized framework and to 455 ms for the personalized framework. This substantial overhead stems from the fact that each HE operation is several orders of magnitude slower than its plaintext counterpart. Furthermore, the problem is exacerbated when inference is performed on a CPU-only server. The runtime increases drastically—27.57 seconds for the generalized framework and 37.61 seconds for the personalized framework—making real-time service infeasible without GPU acceleration. For more complex tasks requiring deeper or more sophisticated models than our proposed model, the computational overhead introduced by homomorphic encryption may also render real-time service infeasible.

Table 3. Comparison of inference performance between CPU and GPU environments, demonstrating the significant computational overhead of HE and the resulting need for GPU acceleration. The times listed in the CPU and GPU columns represent the inference time in seconds for each respective environment.

Data	**Generalized Model**		**Personalized Model**	
	CPU	GPU	CPU	GPU
WESAD	27.57	0.333	37.61	0.455
Hide-and-Seek	27.57	0.333	37.61	0.455

5.4 Comparison with Other Private Semi-supervised Learning Models

In this section, we compare our MLP-based logistic regression model with models used in prior emotion recognition studies to justify our choice of model architecture. We conducted a quantitative comparison with a recent privacy-preserving emotion recognition method by re-implementing the approach of Tsouvalas et al.'s research [68] under the same LOSO (Leave-One-Subject-Out) setting. We conducted experiments using the WESAD dataset, where the number of devices K was set to 14 (excluding the test subject). The training process was performed over $R = 100$ rounds, with each round involving $q = 0.8$ fraction of devices randomly selected to participate. Each selected device performed local training for $E = 1$ step using its own data. In terms of data composition, each device used $L = 0.1$ (10%) of its local dataset as labeled samples, while the remaining data was treated as unlabeled. The influence of the unlabeled data during training was controlled by a weighting parameter $\beta = 1$, and temperature scaling with $T = 2$ was applied to soften the softmax outputs. Predictions were filtered based on a confidence threshold τ, which ranged from 0.5 to 0.9 depending on the formulation. Pseudo-labels were then generated for confident predictions, and both labeled and pseudo-labeled samples were used to compute the total loss during local training. After local updates, the global model was aggregated using weighted averaging across participating devices.

Our training strategy consists of an initial supervised phase to train a shared global model, followed by a personalized semi-supervised co-training phase. For comparison, we refer to baseline [68], which trains a

federated model using both labeled (10%) and unlabeled (90%) data in a mixed supervised and semi-supervised manner. In terms of performance, the baseline [68] achieved an average accuracy of 0.6730 with a standard deviation of 0.0066 across five runs. Our proposed method yielded an accuracy of 0.966 in the general phase and improved further to 0.978 after personalization. These results demonstrate that our approach provides higher accuracy and stability even in a pseudo-label-based learning setup.

While the numerical comparison shows that our method significantly outperforms Tsouvalas et al.'s model in terms of accuracy, we believe the more fundamental difference lies in the level of security guaranteed. Tsouvalas et al.'s method adopts federated learning to enhance privacy, where raw data is not sent to the server, and only local model parameters are shared. However, this does not guarantee security in a strict cryptographic sense, as numerous studies have shown that sensitive information can be reconstructed from local model updates. In contrast, our method achieves a strong cryptographic guarantee by employing homomorphic encryption (HE), which satisfies the IND-CPA security standard—a formal and rigorous definition of semantic security. Therefore, we believe the most meaningful distinction between our work and Tsouvalas et al.'s work lies not merely in the numerical performance gap, but in the fundamentally different security guarantees provided.

6 DISCUSSION

6.1 Theoretical Contribution from Usable Security Perspective

As wearable devices and smart home technologies become more widespread, awareness of and attention to privacy issues have grown significantly [4, 24, 32, 71, 73]. The concept of *usable security* is not just about making systems more secure, but also about ensuring that users can understand, trust, and actually use those security mechanisms in real-world [74]. In other words, security systems that directly interact with human users must be evaluated not only based on their technical rigor and robustness, but also in terms of user experience, cognitive load, and the degree of trust they foster between users and the technology. Previous studies have pointed out that users may struggle to interact with complex encryption or authentication mechanisms and that such complexity often results in the avoidance of security technologies [6, 31].

In this context, we noted that the EER framework is not only technically rigorous but also practically usable, that is well-suited for real-world environments. First, HE is considered one of the most secure privacy-preserving techniques, as it enables computations without exposing raw data [29]. As raw data are collected on the local device and transmitted to the server in an encrypted form, the risk of data reconstruction or exposure of information that could be inferred from the raw data is significantly reduced. In addition, the EER framework is designed to operate without requiring users to understand or manually configure complex security mechanisms. Recent studies on users' perceptions of devices that collect private data reported that users are often unaware of the privacy risks associated with the services they are using [5, 27, 42, 64]. Although some users expressed concern that sensitive information might be inferred from the collected data, they had limited awareness of how accurate such inferences could be or what types of information could potentially be inferred [70]. Moreover, some users unintentionally granted unnecessary privacy permissions to use the service, exposing themselves to additional risks [72]. These findings suggest that in real-world environments, individuals may vary in their understanding of technology and sensitivity to privacy. Therefore, security systems should aim to minimize users' cognitive burden while preserving the core functionality of the service. In this regard, the EER framework offers a significant advantage in real-world contexts. By enabling sensitive data to be processed in an encrypted form throughout the process, it minimizes users' cognitive burden as they are not required to comprehend or manage complex privacy configurations. As a result, we note that the EER framework provides a practical solution for delivering emotion recognition services that are both secure and usable in real-world environments.

Beyond its technical design, the contribution of EER can be further contextualized through a recent theoretical taxonomy of Privacy-Enhancing Technologies (PETs) in the ubiquitous computing domain. O'Hagan et al. [56]

classified PETs into four categories (i.e., access control, data obfuscation, consent, and situational awareness) and proposed a framework that further distinguishes these technologies by the protection target (i.e., user vs. bystander) and the mode of operation (i.e., intrinsic vs. extrinsic). The EER system falls under *user-level intrinsic data obfuscation*, as it *protects the user's own data (i.e., user-level)* and *operates entirely within the system (i.e., intrinsic)* without any external intervention. We note that the EER framework provides empirical validation of this classification scheme, demonstrating how user-level intrinsic data obfuscation can be effectively implemented in practice.

6.2 Multimodal, Semi-supervised and Personalized Emotion Recognition

In this study, we used a publicly available multimodal dataset (i.e., Hide-and-seek) to evaluate the proposed framework. However, we note that the EER framework is not limited to a specific type of dataset and can be extended to incorporate additional data modalities. Stress/emotion recognition can additionally utilize other types of data, such as image data like facial expressions, eye gaze, and body posture [2, 44, 45, 59], text data like social media posts, emails, and chat logs [49, 52], behavioral data such as mouse and keyboard inputs, smartphone usage patterns, and GPS location data [19, 51, 53, 55], as well as environmental data like temperature, humidity, and lighting [53, 55, 76]. Since the proposed EER framework is not limited to a specific type of data, it has the potential for expansion across various industries. While we used commonly utilized stress/emotion recognition datasets and a dataset based on a call center workplace scenario, the EER framework can be extended to fields where data security is critical, such as healthcare, education, and customer service.

In addition, this study further enhanced the practicality of the EER framework by showing that it can also be applied to semi-supervised learning. Previous studies on stress/emotion recognition have predominantly relied on data labeled under experimental conditions or self-reported by participants. As a result, a common limitation has been the difficulty of collecting data and obtaining additional labels from new users when applying models to them [20, 50, 58]. However, in this study, we demonstrated that encryption through HE could also be utilized for unsupervised learning, thereby enabling the application of the model to new users without requiring labeled data. This highlights the high applicability of the proposed framework.

Another issue often raised in stress/emotion recognition research is model personalization. Physiological data commonly used in stress/emotion recognition are known to exhibit significant individual differences [52, 53, 55]. Therefore, even if a central model is trained on data from diverse users, it is necessary to personalize the model based on specific user data to enhance the model performance. In this study, we demonstrated that the model's performance could be further improved by personalizing it through co-training, and that process could also be conducted using encrypted data through HE. By proving that the essential processes required for applying stress/emotion recognition models to real-world scenarios can be performed in an encrypted state, we not only strengthened data security but also enhanced the applicability of the models.

6.3 Limitations

We propose the EER framework, which enables emotion and stress detection using HE without privacy concerns. However, our proposed framework has several limitations.

The first limitation concerns the additional burden on the client side, both in terms of encryption/decryption latency and increased communication time due to large ciphertext sizes. In the proposed framework, the HE-based inference process involves the following steps: (1) Encryption on the user device → (2) Transmission to the server → (3) Computation on the server → (4) Return transmission to the user → (5) Decryption on the user device. Based on CKKS benchmarks reported in [40] for a Raspberry Pi 4, we estimate the following timings: Encryption on the client device takes approximately 16 ms for a 4 MB ciphertext. Transmission to the server takes around 32 ms, assuming a 1 Gbps network bandwidth. This duration may increase in low-bandwidth settings.

Server-side inference requires approximately 455 ms, based on a GPU-equipped server with 24 GB of memory. Transmission back to the client incurs another 32 ms. Decryption on the client takes approximately 21 ms. While these additional delays are negligible relative to the overall runtime and do not hinder real-time performance in typical settings, they may become non-trivial in constrained environments—for example, when network bandwidth is limited or the user's device has low computational capabilities. In such cases, the perceived latency by the end-user could increase, which is an important consideration for deployment in real-world scenarios.

The second limitation is the challenges in training models with encrypted data. In traditional deep learning training, researchers typically monitor the loss curve and compare input data with predictions to assess the model's learning progress. However, in an HE environment, direct access to data is not possible, making it difficult to verify whether the model is functioning correctly during training. This restriction prevents the use of early stopping, which is commonly employed to halt training when performance no longer improves. Consequently, determining an appropriate number of epochs is crucial to prevent overfitting or underfitting, which can significantly impact model performance. Additionally, since model weights cannot be directly inspected during training, ensuring proper learning can be challenging. Despite these challenges, our study demonstrated that incorporating unlabeled subject data for training improved both accuracy and F1-score, suggesting that training with encrypted data did not cause a significant performance degradation. Moreover, these results indicate the potential of personalized modeling using HE, further supporting its feasibility in privacy-preserving stress and emotion detection.

7 CONCLUSION

In this paper, we suggested an EER framework by employing holomorphic encryption to enable model training and inference on encrypted data. To evaluate the framework, we utilized publicly available datasets WESAD and Hide-and-seek to validate the model performance. Through the experiment, we demonstrated that the MLP-based linear regression model can infer stress/emotion states using encrypted data with performance comparable to when inferring based on plaintext data. Additionally, we showed that the MLP-based linear regression model can be personalized using encrypted unlabeled data with the model performance improvement. Consequently, we demonstrated that HE enables data encryption in a two-party setting, user and server, effectively addressing privacy concerns in stress/emotion recognition research. Since our proposed framework does not involve a third party, the risk of data leakage due to third-party involvement is also reduced. The EER framework is particularly useful in environments where privacy concerns are critical, such as healthcare and workplace settings.

Acknowledgments

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.RS-2024-00398360, Development of a user-friendly and efficiency-optimized real-time homomorphic statistical analysis processing platform, 25% & No.RS-2024-00399401, Development of post-quantum security infrastructure transition and comprehensive quantum security verification technology, 25%), the Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government (MSIT) (25ZB1200, Fundamental Technology Research for Human-Centric Autonomous Intelligent Systems, 25%), and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-20253019, A Framework for Modeling High-Level Interactions for Automated Analysis of Collective Intelligence in eSports Team, 25%).

References

[1] 2024. Lattigo v6. Online: https://github.com/tuneinsight/lattigo. EPFL-LDS, Tune Insight SA.

[2] Ahmed Abdou, Ekta Sood, Philipp Müller, and Andreas Bulling. 2022. Gaze-enhanced crossmodal embeddings for emotion recognition. *Proceedings of the ACM on Human-Computer Interaction* 6, ETRA (2022), 1–18.

[3] Aikata Aikata and Sujoy Sinha Roy. 2024. Secure and Efficient Outsourced Matrix Multiplication with Homomorphic Encryption. In *International Conference on Cryptology in India*. Springer, 51–74.
[4] Nada Alhirabi, Stephanie Beaumont, Jose Tomas Llanos, Dulani Meedeniya, Omer Rana, and Charith Perera. 2023. PARROT: Interactive privacy-aware internet of things application design tool. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 1 (2023), 1–37.
[5] Abdulmajeed Alqhatani and Heather Richter Lipford. 2019. {"There} is nothing that I need to keep {secret"}: Sharing Practices and Concerns of Wearable Fitness Data. In *Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019)*. 421–434.
[6] Joseph Bonneau, Cormac Herley, Paul C Van Oorschot, and Frank Stajano. 2012. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *2012 IEEE symposium on security and privacy*. IEEE, 553–567.
[7] Brandon M Booth, Tiantian Feng, Abhishek Jangalwa, and Shrikanth S Narayanan. 2019. Toward robust interpretable human movement pattern analysis in a workplace setting. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 7630–7634.
[8] Jean-Philippe Bossuat, Christian Mouchet, Juan Troncoso-Pastoriza, and Jean-Pierre Hubaux. 2021. Efficient bootstrapping for approximate homomorphic encryption with non-sparse keys. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 587–617.
[9] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. 2014. (Leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)* 6, 3 (2014), 1–36.
[10] Zvika Brakerski and Vinod Vaikuntanathan. 2011. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *Annual cryptology conference*. Springer, 505–524.
[11] Zvika Brakerski and Vinod Vaikuntanathan. 2014. Efficient fully homomorphic encryption from (standard) LWE. *SIAM Journal on computing* 43, 2 (2014), 831–871.
[12] Ruizhi Cheng, Yuetong Wu, Ashish Kundu, Hugo Latapie, Myungjin Lee, Songqing Chen, and Bo Han. 2024. MetaFL: Privacy-preserving User Authentication in Virtual Reality with Federated Learning. In *Proceedings of the 22nd ACM Conference on Embedded Networked Sensor Systems*. 54–67.
[13] Jung Hee Cheon, Kyoohyung Han, Andrey Kim, Miran Kim, and Yongsoo Song. 2019. A full RNS variant of approximate homomorphic encryption. In *Selected Areas in Cryptography–SAC 2018: 25th International Conference, Calgary, AB, Canada, August 15–17, 2018, Revised Selected Papers 25*. Springer, 347–368.
[14] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. 2017. Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*. Springer, 409–437.
[15] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. 2017. On Matrix Multiplication with Homomorphic Encryption. In *International Conference on Information Security and Cryptology*. Springer, 73–84.
[16] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. 2018. Better Bootstrapping for Approximate Homomorphic Encryption. In *International Conference on Cryptology and Information Security in Latin America*. Springer, 3–16.
[17] Prateek Chhikara, Prabhjot Singh, Rajkumar Tekchandani, Neeraj Kumar, and Mohsen Guizani. 2020. Federated learning meets human emotions: A decentralized framework for human–computer interaction for IoT applications. *IEEE Internet of Things Journal* 8, 8 (2020), 6949–6962.
[18] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachene. 2016. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In *Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I 22*. Springer, 3–33.
[19] Vedant Das Swain, Koustuv Saha, Hemang Rajvanshy, Anusha Sirigiri, Julie M Gregg, Suwen Lin, Gonzalo J Martinez, Stephen M Mattingly, Shayan Mirjafari, Raghu Mulukutla, et al. 2019. A multisensor person-centered approach to understand the role of daily activities in job performance with organizational personas. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 4 (2019), 1–27.
[20] Elena Di Lascio, Shkurta Gashi, Juan Sebastian Hidalgo, Beatrice Nale, Maike E Debus, and Silvia Santini. 2020. A multi-sensor approach to automatically recognize breaks and work activities of knowledge workers in academia. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 3 (2020), 1–20.
[21] Léo Ducas and Daniele Micciancio. 2015. FHEW: bootstrapping homomorphic encryption in less than a second. In *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 617–640.
[22] Fatima Elhattab, Sara Bouchenak, and Cédric Boscher. 2024. Pastel: Privacy-preserving federated learning in edge computing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 4 (2024), 1–29.
[23] Fatima Elhattab, Sara Bouchenak, Rania Talbi, and Vlad Nitu. 2023. Robust federated learning for ubiquitous computing through mitigation of edge-case backdoor attacks. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 4 (2023), 1–27.

[24] Lihua Fan, Shuning Zhang, Yan Kong, Xin Yi, Yang Wang, Xuhai" Orson" Xu, Chun Yu, Hewu Li, and Yuanchun Shi. 2024. Evaluating the Privacy Valuation of Personal Data on Smartphones. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 3 (2024), 1–33.

[25] Tiantian Feng and Shrikanth S Narayanan. 2020. Modeling behavioral consistency in large-scale wearable recordings of human bio-behavioral signals. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 1011–1015.

[26] Glenn J Fernandes, Jiayi Zheng, Mahdi Pedram, Christopher Romano, Farzad Shahabi, Blaine Rothrock, Thomas Cohen, Helen Zhu, Tanmeet S Butani, Josiah Hester, et al. 2024. HabitSense: A Privacy-Aware, AI-Enhanced Multimodal Wearable Platform for mHealth Applications. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 3 (2024), 1–48.

[27] Sandra Gabriele and Sonia Chiasson. 2020. Understanding fitness tracker users' security and privacy knowledge, attitudes and behaviours. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–12.

[28] Neha Gahlan and Divyashikha Sethia. 2024. Federated learning inspired privacy sensitive emotion recognition based on multi-modal physiological sensors. *Cluster Computing* 27, 3 (2024), 3179–3201.

[29] Craig Gentry. 2009. *A fully homomorphic encryption scheme*. Stanford university.

[30] Thore Graepel, Kristin Lauter, and Michael Naehrig. 2012. Using Fully Homomorphic Encryption for Statistical Analysis of Categorical, Ordinal and Numerical Data. *IACR Cryptology ePrint Archive* 2012 (2012), 441.

[31] Cormac Herley. 2009. So long, and no thanks for the externalities: the rational rejection of security advice by users. In *Proceedings of the 2009 workshop on New security paradigms workshop*. 133–144.

[32] Changshuo Hu, Xiao Ma, Xinger Huang, Yiran Shen, and Dong Ma. 2024. LR-Auth: Towards Practical Implementation of Implicit User Authentication on Earbuds. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 4 (2024), 1–27.

[33] Dhruv Jain, Khoa Huynh Anh Nguyen, Steven M. Goodman, Rachel Grossman-Kahn, Hung Ngo, Aditya Kusupati, Ruofei Du, Alex Olwal, Leah Findlater, and Jon E. Froehlich. 2022. Protosound: A personalized and scalable sound recognition system for deaf and hard-of-hearing users. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–16.

[34] Mimansa Jaiswal and Emily Mower Provost. 2020. Privacy enhanced multimodal neural representations for emotion recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 7985–7993.

[35] Houtan Jebelli, Byungjoo Choi, and SangHyun Lee. 2019. Application of wearable biosensors to construction sites. I: Assessing workers' stress. *Journal of Construction Engineering and Management* 145, 12 (2019), 04019079.

[36] Xiaoqian Jiang, Miran Kim, Kristin Lauter, and Yongsoo Song. 2018. Secure outsourced matrix computation and application to neural networks. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 1209–1222.

[37] Samira Ebrahimi Kahou, Xavier Bouthillier, Pascal Lamblin, Caglar Gulcehre, Vincent Michalski, Kishore Konda, Sébastien Jean, Pierre Froumenty, Yann Dauphin, Nicolas Boulanger-Lewandowski, et al. 2016. Emonets: Multimodal deep learning approaches for emotion recognition in video. *Journal on Multimodal User Interfaces* 10, 2 (2016), 99–111.

[38] Ayanga Imesha Kumari Kalupahana, Ananta Narayanan Balaji, Xiaokui Xiao, and Li-Shiuan Peh. 2023. SeRaNDiP: Leveraging Inherent Sensor Random Noise for Differential Privacy Preservation in Wearable Community Sensing Applications. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 2 (2023), 1–38.

[39] Mina Khan, Glenn Fernandes, Akash Vaish, Mayank Manuja, and Pattie Maes. 2021. Wearable system for personalized and privacy-preserving egocentric visual context detection using on-device deep learning. In *Adjunct Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization*. 35–40.

[40] Het Khatusuriya, Dhvani Patel, and Martin Parmar. 2025. Benchmarking Homomorphic Encryption on Low-Power Devices: Trade-offs Between PHE and FHE. (2025).

[41] Eunsang Lee, Joon-Woo Lee, Jong-Seon No, and Young-Sik Kim. 2021. Minimax Approximation of Sign Function by Composite Polynomial for Homomorphic Comparison. *IEEE Transactions on Dependable and Secure Computing* (2021).

[42] Hyunsoo Lee, Soowon Kang, and Uichin Lee. 2022. Understanding privacy risks and perceived benefits in open dataset collection for mobile affective computing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 2 (2022), 1–26.

[43] Joon-Woo Lee, Eunsang Lee, Yongwoo Lee, Young-Sik Kim, and Jong-Seon No. 2021. High-precision bootstrapping of RNS-CKKS homomorphic encryption using optimal minimax polynomial approximation and inverse sine function. In *Advances in Cryptology–EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part I 40*. Springer, 618–647.

[44] Sze Chit Leong, Yuk Ming Tang, Chung Hin Lai, and CKM Lee. 2023. Facial expression and body gesture emotion recognition: A systematic review on the use of visual data in affective computing. *Computer Science Review* 48 (2023), 100545.

[45] Shan Li and Weihong Deng. 2020. Deep facial expression recognition: A survey. *IEEE transactions on affective computing* 13, 3 (2020), 1195–1215.

[46] Xiaochen Li, Sicong Liu, Zimu Zhou, Bin Guo, Yuan Xu, and Zhiwen Yu. 2024. EchoPFL: Asynchronous Personalized Federated Learning on Mobile Devices with On-Demand Staleness Control. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 1 (2024), 1–22.

[47] Jian Liao, Adnan Karim, Shivesh Singh Jadon, Rubaiat Habib Kazi, and Ryo Suzuki. 2022. RealityTalk: Real-time speech-driven augmented presentation for AR live storytelling. In *Proceedings of the 35th annual ACM symposium on user interface software and technology*. 1–12.

[48] Peter Mantello, Manh-Tung Ho, Minh-Hoang Nguyen, and Quan-Hoang Vuong. 2023. Bosses without a heart: socio-demographic and cross-cultural determinants of attitude toward Emotional AI in the workplace. *AI & society* 38, 1 (2023), 97–119.

[49] Gloria Mark, Shamsi T Iqbal, Mary Czerwinski, Paul Johns, Akane Sano, and Yuliya Lutchyn. 2016. Email duration, batching and self-interruption: Patterns of email use on productivity and stress. In *Proceedings of the 2016 CHI conference on human factors in computing systems*. 1717–1728.

[50] Daniel McDuff, Eunice Jun, Kael Rowan, and Mary Czerwinski. 2019. Longitudinal observational evidence of the impact of emotion regulation strategies on affective expression. *IEEE Transactions on Affective Computing* 12, 3 (2019), 636–647.

[51] Shayan Mirjafari, Kizito Masaba, Ted Grover, Weichen Wang, Pino Audia, Andrew T Campbell, Nitesh V Chawla, Vedant Das Swain, Munmun De Choudhury, Anind K Dey, et al. 2019. Differentiating higher and lower job performers in the workplace using mobile sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 2 (2019), 1–24.

[52] Mehrab Bin Morshed, Javier Hernandez, Daniel McDuff, Jina Suh, Esther Howe, Kael Rowan, Marah Abdin, Gonzalo Ramos, Tracy Tran, and Mary Czerwinski. 2022. Advancing the understanding and measurement of workplace stress in remote information workers from passive sensors and behavioral data. In *2022 10th International Conference on Affective Computing and Intelligent Interaction (ACII)*. IEEE, 1–8.

[53] Mehrab Bin Morshed, Koustuv Saha, Richard Li, Sidney K D'Mello, Munmun De Choudhury, Gregory D Abowd, and Thomas Plötz. 2019. Prediction of mood instability with passive sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 3 (2019), 1–21.

[54] Vansh Narula, Kexin Feng, and Theodora Chaspari. 2020. Preserving privacy in image-based emotion recognition through user anonymization. In *Proceedings of the 2020 International Conference on Multimodal Interaction*. 452–460.

[55] Subigya Nepal, Gonzalo J Martinez, Shayan Mirjafari, Stephen Mattingly, Vedant Das Swain, Aaron Striegel, Pino G Audia, and Andrew T Campbell. 2021. Assessing the impact of commuting on workplace performance using mobile sensing. *IEEE Pervasive Computing* 20, 4 (2021), 52–60.

[56] Joseph O'Hagan, Pejman Saeghe, Jan Gugenheimer, Daniel Medeiros, Karola Marky, Mohamed Khamis, and Mark McGill. 2023. Privacy-enhancing technology and everyday augmented reality: Understanding bystanders' varying needs for awareness and consent. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 4 (2023), 1–35.

[57] Eunji Park, Yugyeong Jung, Inyeop Kim, and Uichin Lee. 2023. Charlie and the semi-automated factory: data-driven operator behavior and performance modeling for human-machine collaborative systems. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–16.

[58] Eunji Park, Duri Lee, Yunjo Han, James Diefendorff, and Uichin Lee. 2024. Hide-and-seek: Detecting Workers' Emotional Workload in Emotional Labor Contexts Using Multimodal Sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 3 (2024), 1–28.

[59] Phuong Pham and Jingtao Wang. 2017. Understanding emotional responses to mobile video advertisements via physiological signal sensing and facial expression analysis. In *Proceedings of the 22nd International Conference on intelligent user interfaces*. 67–78.

[60] Alessandro Pogliaghi, Elena Di Lascio, Shkurta Gashi, Emanuela Piciucco, Silvia Santini, and Martin Gjoreski. 2022. Multi-task learning for stress recognition. In *Adjunct Proceedings of the 2022 ACM International Joint Conference on Pervasive and Ubiquitous Computing and the 2022 ACM International Symposium on Wearable Computers*. 202–206.

[61] Kat Roemmich, Florian Schaub, and Nazanin Andalibi. 2023. Emotion AI at work: Implications for workplace surveillance, emotional labor, and emotional privacy. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–20.

[62] Florian Schaule, Jan Ole Johanssen, Bernd Bruegge, and Vivian Loftness. 2018. Employing consumer wearables to detect office workers' cognitive load for interruption management. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies* 2, 1 (2018), 1–20.

[63] Philip Schmidt, Attila Reiss, Robert Duerichen, Claus Marberger, and Kristof Van Laerhoven. 2018. Introducing wesad, a multimodal dataset for wearable stress and affect detection. In *Proceedings of the 20th ACM international conference on multimodal interaction*. 400–408.

[64] Stefan Schneegass, Romina Poguntke, and Tonja Machulla. 2019. Understanding the impact of information representation on willingness to share information. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–6.

[65] Mauricio Soto, Chris Satterfield, Thomas Fritz, Gail C Murphy, David C Shepherd, and Nicholas Kraft. 2021. Observing and predicting knowledge worker stress, focus and awakeness in the wild. *International Journal of Human-Computer Studies* 146 (2021), 102560.

[66] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. 2022. Towards personalized federated learning. *IEEE transactions on neural networks and learning systems* 34, 12 (2022), 9587–9603.

[67] Brian Testa, Yi Xiao, Harshit Sharma, Avery Gump, and Asif Salekin. 2023. Privacy against real-time speech emotion detection via acoustic adversarial evasion of machine learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 3 (2023), 1–30.

[68] Vasileios Tsouvalas, Tanir Ozcelebi, and Nirvana Meratnia. 2022. Privacy-preserving speech emotion recognition through semi-supervised federated learning. In *2022 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. IEEE, 359–364.

[69] Terumi Umematsu, Akane Sano, Sara Taylor, Masanori Tsujikawa, and Rosalind W Picard. 2020. Forecasting stress, mood, and health from daytime physiology in office workers and students. In *2020 42nd annual international conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 5953–5957.

[70] Lev Velykoivanenko, Kavous Salehzadeh Niksirat, Noé Zufferey, Mathias Humbert, Kévin Huguenin, and Mauro Cherubini. 2021. Are those steps worth your privacy? Fitness-tracker users' perceptions of privacy and utility. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 4 (2021), 1–41.

[71] Chixiang Wang, Weijia He, Timothy J Pierson, and David Kotz. 2024. Moat: Adaptive Inside/Outside Detection System for Smart Homes. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 4 (2024), 1–31.

[72] Xuetao Wei, Lorenzo Gomez, Iulian Neamtiu, and Michalis Faloutsos. 2012. Permission evolution in the android ecosystem. In *Proceedings of the 28th Annual Computer Security Applications Conference*. 31–40.

[73] Ben Weinshel, Yuvraj Agarwal, and Lujo Bauer. 2025. "I would still use it but I wouldn't trust it": Evaluating Mechanisms for Transparency and Control for Smart-Home Sensors. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 9, 2 (2025), 1–33.

[74] Alma Whitten and J Doug Tygar. 1999. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0.. In *USENIX security symposium*, Vol. 348. 169–184.

[75] Mengwei Xu, Feng Qian, Qiaozhu Mei, Kang Huang, and Xuanzhe Liu. 2018. Deeptype: On-device deep learning for input personalization service with minimal privacy concern. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 4 (2018), 1–26.

[76] Jing Nathan Yan, Ziwei Gu, and Jeffrey M Rzeszotarski. 2021. Tessera: Discretizing data analysis workflows on a task level. In *Proceedings of the 2021 CHI conference on human factors in computing systems*. 1–15.

[77] Tuo Zhang, Lei Gao, Chaoyang He, Mi Zhang, Bhaskar Krishnamachari, and A Salman Avestimehr. 2022. Federated learning for the internet of things: Applications, challenges, and opportunities. *IEEE Internet of Things Magazine* 5, 1 (2022), 24–29.

[78] Weibin Zhang, Youpeng Li, Lingling An, Bo Wan, and Xuyu Wang. 2024. SARS: A Personalized Federated Learning Framework Towards Fairness and Robustness against Backdoor Attacks. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 4 (2024), 1–24.

[79] Wei Zhang, Xiaoyun Wang, and Zheng Liu. 2024. Homomorphic Matrix Operations under Bicyclic Encoding. *IACR Cryptology ePrint Archive* 2024 (2024), 1762.

[80] Yulei Zhang, Jianfeng Ma, Ximeng Li, Yichen Zhang, and Qiaoyan Liu. 2019. Secure Outsourced Computation of Multiple Matrix Multiplication Based on Fully Homomorphic Encryption. *IEEE Transactions on Cloud Computing* (2019).

[81] Yulei Zhang, Jianfeng Ma, Ximeng Li, Yichen Zhang, and Qiaoyan Liu. 2022. Secure Matrix Multiplication Based on Fully Homomorphic Encryption. *The Journal of Supercomputing* 78 (2022), 1425–1445.

[82] Yulei Zhang, Jianfeng Ma, Ximeng Li, Yichen Zhang, and Qiaoyan Liu. 2023. Secure Outsourced Matrix Multiplication with Fully Homomorphic Encryption. In *International Conference on Information Security*. Springer, 195–210.

[83] Tongqing Zhou, Zhiping Cai, Bin Xiao, Leye Wang, Ming Xu, and Yueyue Chen. 2018. Location privacy-preserving data recovery for mobile crowdsensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 3 (2018), 1–23.

[84] Manuela Züger, Sebastian C Müller, André N Meyer, and Thomas Fritz. 2018. Sensing interruptibility in the office: A field study on the use of biometric and computer interaction sensors. In *Proceedings of the 2018 CHI conference on human factors in computing systems*. 1–14.

A Extended Preliminaries

A.1 Details in Homomorphic Encryption

Among the operations described above, PMult and CMult require additional considerations. In HE, real values are not stored directly in the ciphertext; instead, they are represented as integers scaled by a large scale factor. When a multiplication operation is performed, this scale factor is squared, which can affect subsequent computations. Therefore, after performing PMult or CMult operations, a rescaling step is necessary to restore the scale to its original value. During this rescaling process, the ciphertext size decreases by the bit length of the scale factor. Since the ciphertext size reduces with each multiplication, the number of multiplications that can be performed is inherently limited. The maximum number of allowed multiplications for a ciphertext is referred to as its *level*. When two ciphertexts with levels ℓ_1 and ℓ_2 undergo CMult, the resulting ciphertext level is $\min\{\ell_1, \ell_2\} - 1$.

Similarly, when a ciphertext with level ℓ undergoes PMult, its resulting level is $\ell - 1$. The *depth* of a computational circuit is defined as the difference between the input ciphertext level and the output ciphertext level after executing a sequence of operations. If additional multiplications need to be performed on a ciphertext at level 0, a special operation called *bootstrapping* can be applied to restore the ciphertext's level while preserving its data. However, bootstrapping is computationally expensive, making it crucial to minimize the number of bootstrapping operations. Consequently, one of the key objectives in designing HE circuits is to minimize circuit depth, thereby reducing overall computational cost. The multiplication operation consumes $Depth$. Once a ciphertext has exceed the depth, no further multiplication operations are possible, and bootstrapping operations must be performed to restore the ciphertext's depth.

Additionally, there is a rotation operation $rot(r)$ that rotates the elements of the vector. Rotation and multiplication require a rotation key and an evaluation key, denoted $rtk(r)$ and evk, which are generated on the client in the $KeyGen()$ procedure along with the public key and passed to the server. The formal definition of each parameter and operation is given follow.

- N(slot size) : The length of the plaintext vector corresponding to each ciphertext.
- Level: The number of multiplicative operations the current ciphertext can perform. As multiplication is performed, the level decreases and when it becomes zero, no more multiplication can be performed.
- Depth : The upper limit of levels a ciphertext can have. That is, the level if no multiplications were ever performed. The size of the Depth is determined by the security parameters of the scheme.
- KeyGen$(sk, \mathcal{G})$:
  1. Generate public key pk from sk.
  2. Generate evaluation key evk from sk
  3. Generate rotation keys $rtk(i)$ from sk, for each i in $\mathcal{G}$.
- Let ct_1, ct_2 be the ciphertext of vectors v_1, v_2
  $Add(ct_1, ct_2) = ct_{Add}$: Decryption vector of ct_{Add} is $v_1 \oplus v_2$.
  $Mult(ct_1, ct_2, evk) = ct_{mul}$: Decryption vector of ct_{mul} is $v_1 \otimes v_2$, And its level is $\min\{\mathrm{Lv}(ct_1), \mathrm{Lv}(ct_2)\} - 1$.
  $Rot(ct_1, rtk(r)) = ct_{rot}$: Decryption vector of ct_r is $\rho(v_1, r) = (v_r, v_{r+1}, \cdots, v_{N-1}.v_0, \cdots, v_{r-1})$.

A.2 Homomorphic Encryption with AI models

The two main operations required for MLP are matrix multiplication and activation function evaluation. Here, we only sketch the algorithm for performing matrix multiplication and activation function using the four basic operations described in the previous section, with more details provided in the algorithm in the appendix.

The standard method for evaluating activation functions such as Sigmoid and ReLU in HE is polynomial approximation. The server generates an approximate polynomial for the activation function using an approximation algorithm by setting the approximation interval and approximation degree. It should be noted that the approximation interval must be set very conservatively because the server does not know the ciphertext value. Subsequently, the server can evaluate the polynomial using the coefficients of the generated polynomial, the ciphertext, and the Add, PMult, CMult operations. Additionally, various optimization methods have been studied to minimize the number of CMult operations requiring *key-switching* in polynomial evaluation[8, 16, 41, 43]. We used the polynomial evaluation algorithm proposed by [8] which is a divide-and-conquer algorithm that divides an approximation polynomial into fixed base polynomials (suitable for approximation) to construct a quotient polynomial and a remainder polynomial. A detailed description is given in Algorithm 1 in the Appendix.

Similar to the approximation of activation functions, significant progress has also been made in researching matrix operations, which are central to machine learning [3, 15, 30, 79–82]. Considering that matrix addition involves adding elements with the same index and matrix multiplication involves multiple inner product operations, it is clear that matrix addition and multiplication can be constructed using the basic operations add, PMult, CMult, Rot.

However, while matrix addition can be performed relatively simply in a HE environment, the naive matrix multiplication described earlier incurs high computational costs due to frequent *key-switching* operations. We follow the optimized matrix multiplication algorithm [36] and describe the necessary algorithms in Algorithms 2 and 3 in the appendix. The main idea of this algorithm is that matrix multiplication can be decomposed into permutation, element-wise multiplication, and addition.

B Appendix - Algorithms

Algorithm 1 BSGS Polynomial Evaluation Algorithm

Procedure: EvalPoly(ct, $P(x)$)

Input: ct: ciphertext of x , $P(x) = \sum_{k=0}^{deg-1} c_k \cdot x^k$: target polynomial function for *power* basis.

Output: $ct_{rtn} = Enc(P(x))$: ciphertext of P(x).

// precomputed

$M \leftarrow \lceil \log(\deg+1) \rceil$
$I \leftarrow \lfloor M/2 \rfloor$
$h_1 \leftarrow P(x)$
Find $q(x), r(x)$ s.t. $h_1(x) = q(x) \cdot T_{2^{M-1}} + r(x)$ and $h_2 \leftarrow q(x),\ h_3 \leftarrow r(x)$
To construct the tree Tr, recurse on step 5 by $h_i = h_{2i} \cdot T_{2^t} + h_{2i+1}$ until $t \geq I$, where $t = M - 1 - \lfloor \log i \rfloor$
// evaluation
$T_0 \leftarrow 1, T_1 \leftarrow ct$
for $i = 2$ **to** $2^I - 1$ **do**
    $T_{i=a+b} \leftarrow \text{Add}(2 \cdot \text{CMult}(T_a, T_b, mk), -T_{|a-b|})$
end for
for $i = I$ **to** M **do**
    $T_{2^i} \leftarrow \text{Add}(2 \cdot \text{CMult}(T_{2^{i-1}}, T_{2^{i-1}}, mk), -1)$
end for
Evaluate leaf node of tree Tr using T_i for $0 \leq i < 2^{I-1}$
Evaluate internal node and root node of tree Tr using $h_i = h_{2i} \cdot T_{2^i} + h_{2i+1}$ for $I \leq i < M$.
return ct_{rtn}: Computed ciphertext corresponding to h_1

Algorithm 2 BSGS Linear Transformation Algorithm

Procedure: LT(ct, A)
Input: ct: ciphertext of vector $\mathbf{v}$, $D(A) = \{d_i : d_i[j] = A[i][i+j \mod n]$ for $0 \le i, j < n\}$: list of all diagonal vectors of matrix A, where $A \in \mathbb{R}^{n \times n}$.
Output: $Enc(A \cdot \mathbf{v})$: ciphertext of vector $A \cdot \mathbf{v}$

$n' \leftarrow \lceil \sqrt{n} \rceil$
$ct_{rtn} \leftarrow ct_{zero} = Enc(\mathbf{0})$
for $i = 0$ **to** n' **do**
  giant$[i] \leftarrow \text{Rot}(ct, i \cdot n', rk_{i \cdot n'})$
end for
for $j = 0$ **to** n' **do**
  $ct_{in} \leftarrow ct_{zero}$
  for $i = 0$ **to** n' **do**
    $ct_{\text{tmp}} \leftarrow \text{PMult}(\text{giant}[i], \rho(d_{i \cdot n' + j}, -j))$
    $ct_{in} \leftarrow \text{Add}(ct_{in}, ct_{temp})$
  end for
  $ct_{in} \leftarrow \text{Rot}(ct_{in}, j, rk_j)$
  $ct_{\text{rtn}} \leftarrow \text{Add}(ct_{rtn}, ct_{in})$
end for
return ct_{rtn}

Algorithm 3 rectangular matrix-matrix multplication algorithm

Procedure: MM($ct.A$, $ct.B$)
Input: $ct.A$: ciphertext of matrix $A \in \mathbb{R}^{\ell \times n}$, $ct.B$: ciphertext of matrix $B \in \mathbb{R}^{n \times n}$.
Output: $Enc(C)$: ciphertext of matrix $C = A \cdot B \in \mathbb{R}^{\ell \times n}$.

$ct_{in} \leftarrow ct_{zero}$
for $i = 0$ **to** $\log(n/\ell)$ **do**
  $ct_{in} \leftarrow \text{Add}(ct_{in}, \text{Rot}(ct_{in},\ i \cdot \ell \cdot n \cdot 2^i,\ rk_{i \cdot \ell \cdot n \cdot 2^i})$
end for
$ct.A^{(0)} \leftarrow \text{LT}(ct_{in}, U^{\sigma})$
$ct.B^{(0)} \leftarrow \text{LT}(ct_{in}, U^{\tau})$
for $i = 1$ **to** l **do**
  $ct.A^{(i)} \leftarrow \text{LT}(ct.A^{(0)}, V^i)$
  $ct.B^{(i)} \leftarrow \text{LT}(ct.B^{(0)}, W^i)$
end for
$ct_{rtn} \leftarrow \text{CMult}(ct.A^{(0)}, ct.B^{(0)}, mk)$
for $i = 1$ **to** l **do**
  $ct_{rtn} \leftarrow \text{Add}(ct_{rtn}, \text{CMult}(ct.A^{(i)}, ct.B^{(i)}))$
end for
$ct_{rrtn} \leftarrow ct_{rtn}$
for $i = 0$ **to** $\log(n/\ell)$ **do**
  $ct_{rrtn} \leftarrow \text{Add}(ct_{rrtn}, \text{Rot}(ct.A,\ i \cdot \ell \cdot n \cdot 2^i,\ rk_{i \cdot \ell \cdot n \cdot 2^i})$
end for
return ct_{rrtn}

Algorithm 4 forward Algorithm

Procedure: forward($ct.X$, $ct.W_1$, $ct.W_2$, $P_{sign}(x)$, $P_\sigma(x)$)
Input: $ct.X$: ciphertext of n intput datas $X \in \mathbb{R}^{\ell \times n}$, $ct.W_1, ct.W_2$: ciphertexts of $W_1 \in \mathbb{R}^{32\times \ell}$ and $W_1 \in \mathbb{R}^{1\times 32}$
$P_{sign}(x)$: approximate polynomial function of Sign$(\cdot)$, $P_\sigma(x)$: approximate polynomial function of $\sigma(\cdot)$
Output: $ct.\hat{y}$: ciphertext of Prediction probabilities for input data X.

$ct.z_1 \leftarrow \text{MM}(ct.W_1, ct.X)$ // 1st layer
$ct.a_{sign} \leftarrow \text{EvalPoly}(ct.z_1, P_{sign}(x))$ // store
$ct.a_1 \leftarrow \text{CMult}(\text{Add}(ct.a_{sign}, 0.5), ct.z_1, mk)$ // store
$ct.z_2 \leftarrow \text{MM}(ct.W_2, ct.a_1)$ // 2nd layer
$ct.\hat{y} \leftarrow \text{EvalPoly}(ct.z_2, P_\sigma(x))$
return $ct.\hat{y}$

Algorithm 5 backwoard Algorithm

Procedure: backward($ct.X$, $ct.\hat{y}$,ct, y, $ct.s$, $ct.W_1$, $ct.W_2$)
Input:
$ct.X$: ciphertext of input data $X^T \in \mathbb{R}^{l\times n}$, $ct.y$, $ct.\hat{y}$, $ct.s$: ciphertext of pseudo label, prediction probabilbily, selection value $\in \mathbb{R}^{1\times n}$, $ct.W_1, ct.W_2$: ciphertext of weight $W_1 \in \mathbb{R}^{1\times 32}$, $W_2 \in \mathbb{R}^{32\times l}$,
Output: $ct.dW_1, ct.dW_2$: ciphertext of gradient value for W_1, W_2.
// gradeint for W_2

$dz \leftarrow \text{Add}(ct.\hat{y}, -ct.y)$
$dz \leftarrow \text{CMult}(dz, ct.s, mk)$
$dW_2 \leftarrow \text{CMult}(dz, a1, mk)$
$dW_2 \leftarrow \text{LT}(dW_2, U^t)$
for $i = 0$ **to** $\log(n)$ **do**
    $dW_2 = \text{Add}(dW_2, \text{Rot}(dW_2, 2^i,\ rk_{2^i})$
end for
// gradeint for W_1
$ct.W_1^t \leftarrow \text{LT}(ct.W_1, U^t)$
$ct_{tmp} \leftarrow \text{CMult}(ct.W_1^t, dz, mk)$
$ct_{tmp} \leftarrow \text{CMult}(ct_{tmp}, ct.a_{sign}, mk)$
$ct.dW_1 \leftarrow \text{MM}(ct_{tmp}, ct.X^t)$
return $ct.dW_1,\ ct.dW_2$

Algorithm 6 View Swap Algorithm

Procedure: Swap(ct)
Input: ct : ciphertext of $\mathbf{x} = (a_1, b_1, a_2, b_2, \cdots a_N, b_N)$, $\mathbf{v}$: masking vector $\mathbf{v} = (1, 0, 1, 0, \cdots, 1, 0)$
Output: ct' : swapped ciphertext of $\mathbf{x}' = (b_1, a_1, b_2, a_2, \cdots, b_N, a_N)$

$ct' \leftarrow \text{PMult}(ct, \mathbf{v})$
$ct' \leftarrow \text{Rot}(ct', -1, rk_{-1})$
$ct_r \leftarrow \text{Rot}(ct, 1, rk_1)$
$ct_r \leftarrow \text{PMult}(ct_r, \mathbf{v})$
$ct' \leftarrow \text{Add}(ct', ct_r)$
return ct'

Algorithm 7 co-training Algorithm

Procedure: cotrain($ct.X$, num_round, num_epoch)
Input: $ct.X$: ciphertext of input data, M : pre-trained two-veiw generalized model
Output: M' : Encrypted co-trained model M'
$M' \leftarrow M$
for $j = 0$ **to** num_round **do**
  $ct.\hat{y} \leftarrow M'$.forward($ct.X$, $ct.W_1$, $ct.W_2$, P_{ReLU}, P_σ)
  $ct.y \leftarrow$ EvalPoly(Add($ct.\hat{y}, -0.5$), P_{sign})
  $ct_{tmp} \leftarrow$ Add($ct.\hat{y}, -0.5$)
  $ct_{tmp} \leftarrow$ CMult(ct_{tmp}, ct_{tmp}, mk)
  $ct_{tmp} \leftarrow$ Add($ct_{tmp}, (t - 0.5)^2$)
  $ct.s \leftarrow$ EvalPoly(ct_{tmp}, P_{sign})
  $ct.y \leftarrow$ Swap($ct.y$)
  $ct.s \leftarrow$ Swap($ct.s$)
  cross-veiw filtering
  for $i = 0$ **to** num_epoch **do**
    $ct.\hat{y} \leftarrow M'$.forward($ct.X$, $ct.W_1$, $ct.W_2$, P_{ReLU}, P_σ)
    $ct.dW_1, ct.dW_2 \leftarrow M'$.backward($ct.X, ct.\hat{y}, ct, y, ct.s, ct.W_1, ct.W_2$)
    $M' \leftarrow$ weight update for M'
  end for
end for
return M'