FLCrypt – Secure Federated Learning for Audio Event Classification using Homomorphic Encryption

Kay Fuhrmeister*, Hao Cui†, Artem Yaroshchuk‡ and Thomas Köllmer§
Fraunhofer IDMT
Ilmenau, Germany
Email: *kay.christopher.fuhrmeister@idmt.fraunhofer.de, †hao.cui@idmt.fraunhofer.de, ‡artem.yaroshchuk@idmt.fraunhofer.de,
§thomas.koellmer@idmt.fraunhofer.de

***Abstract*—In this paper, we introduce FLCrypt[1], a library designed to enhance Federated Learning with additional privacy guarantees by applying Fully Homomorphic Encryption to the model aggregation stage, thereby preventing the aggregator from accessing the unencrypted model parameters of the training participants.**

We evaluate our approach by comparing its accuracy against an unencrypted baseline using an audio classification task aimed at distinguishing metal balls with different surfaces, which serves as a proxy for fault detection in industrial sound analysis. Our findings indicate a marginal decrease in accuracy due to applying Fully Homomorphic Encryption, alongside a significant increase in both runtime and memory demands. Our analysis also concludes that runtime increases linearly with the number of model parameters. Our results lead us to affirm the viability of FLCrypt for Federated Learning applications in acoustic sensor networks with elevated security requirements such as sound classification.

***Index Terms*—Homomorphic Encryption, Federated Learning, Security, Privacy, Audio Event Detection, Acoustic Sensor Networks.**

I. Introduction

With the growing desire to leverage the advantages of AI in a various settings the need to collect large amounts of data has increased tremendously. One such setting is sound event detection. Since the audio recordings collected for this purpose can contain sensitive information, this leads to concerns about the storage of said data as well as the privacy of the individuals whose data was collected. Federated Learning (FL) is a machine learning approach designed to address these concerns.

FL is a collaborative learning framework where a single (global) machine learning model is created by the aggregation of several local models from different end devices. This is done on a central server with the use of an aggregation algorithm, for example by averaging the weights of the local models. The training is carried out by the end devices locally and without transmitting potentially sensitive training data to a centralized location, thus, enhancing privacy. In addition, it eliminates the necessity to store large amounts of data in a single location, reducing the risk of data breaches. Further, the operational costs of distributing data are reduced to sharing significantly smaller models while also decreasing the cost of gathering and aggregating said data. Considering the prevalence of Internet of Things (IoT) devices, FL enables collaborative learning for tasks that benefit from a variety of different information sources without compromising the privacy of those sources [10]. As such, the use of privacy enhancing technologies such as homomorphic encryption was identified in [31] as a promising research topic in the context of the Internet of Sounds (IoS). Sound analysis is an area that can greatly benefit from this approach since acoustic sensor networks have to function in variety of environments with different background noises and different sound events occurring with varying levels of frequency while privacy regarding the audio recordings needs to be protected.

Considering legal regulations that protect data privacy such as the General Data Protection Regulation (GDPR), careless handling of audio data is a significant risk for data holders both legally and in terms of public image. Secondly, privacy risks can deter individuals from volunteering their data which can impede data availability. Hence, it is in the interest of data curators as well as data donors that machine learning preserves privacy.

However, leveraging the potential of machine learning while avoiding potential privacy violations is not an easy task [16]. The benefits of FL make it an attractive approach to mitigate privacy risks. Still, even the disclosure of model weights may reveal information about the underlying data to potential adversaries as was shown through membership inference attacks in [30], [36], [6]. Additionally, model inversion attacks have been carried out in [12]. Therefore, FL applications still need to be designed with additional security measures to protect private data. One possibility to improve upon the security of FL is provided by Homomorphic Encryption (HE) which describes encryption techniques that allow computations on encrypted values. This allows for the encryption of model weights and to perform the model aggregation without decrypting them. Thus, the plaintext model is never revealed to the aggregator and information leakage from said local models to the server is prevented. Further, combining FL with HE allows us to forego integrating the encryption into the machine learning algorithms directly which is more computationally efficient while still substantially increasing security. This is an important advantage for applications in sensor networks with

[1]For evaluation access to FLCrypt, please contact the main author or reach us via https://www.idmt.fraunhofer.de/en/contact.html.

limited resources.

A. Outline

The paper is organized as follows. In Section II, we provide the preliminaries of FL and in Section III the preliminaries of HE. In Section III-A we explain possible security threats in the FL setting. In section Section IV, we provide an overview of applications of HE in application oriented FL settings and Section V contains the evaluation of our experiments of applying HE to sound event detection in an FL context using our framework FLCrypt. To evaluate the results, we compare the model accuracy to the unencrypted baseline using two FL strategies, federated averaging (FedAvg) and federated proximal (FedProx). We conclude our evaluation in Section VI and illustrate future research directions.

II. Federated Learning

Federated Learning is a machine learning approach where several models are trained across multiple end devices (clients) and are aggregated by a server into a global model. The aggregated model is then sent back to every client. Instead of exchanging training data, only the model parameters are exchanged. This eliminates the need to store large amounts of data in a single location and lets data holders retain control of said data. It was introduced in 2016 in [23] and has since been applied to variety of applications, including the field of audio classification.

By having access to more data from different sources, FL can mitigate imbalances such as a lack of data for certain classes or bias in the local data sets. That being said, heterogeneous data can also be detrimental to the accuracy of models obtained through FL training [22] and this is a major challenge when applying FL to realistic settings [19]. To address this issue several approaches have been proposed in [29], [37] and [26] that significantly improve FL results in the heterogenous setting compared to standard FL methods. As such, FL presents us with the opportunity to achieve good accuracy and model generalization in a decentralized learning setting.

A. Federated Learning Aggregation Strategies

The FL approach itself does not specify a particular aggregation method. In this paper, we analyze the Federated Averaging (FedAvg) and the Federated Proximal (FedProx). The FedAvg algorithm, proposed in [23], computes a weighted average of individual model parameters to produce a final global model. The local model parameters are weighted with respect to the client's proportion of the data to ensure that the local model's impact is proportional to the amount of information it contributes to the training process. However, the final aggregated model might not perform as well for clients that contribute comparatively little data.

The FedProx algorithm was introduced in [20] in order to improve upon FedAvg for hardware and data heterogeneity across clients. It does so by introducing an adaptive regularization term on the clients' loss functions that favors local model parameters that are close to the global model in each round of FL. The larger it is, more closeness to the global model will be incentivized. The server-side aggregation of the clients' model parameters stays the same as in FedAvg. Previously, FL was applied to sound event detection for domestic and urban sound monitoring [18] and to audio classification for key word spotting and urban sound classification [14] without the use of additional privacy enhancing technologies. There have also been experiments of using wireless acoustic sensor network in combination with machine learning for vehicle classification ([33], [24]), yielding promising results. Utilizing wireless acoustic networks is of great interest for environmental noise monitoring and audio event detection, especially in the context of Smart Cities [1]. However, to the best of our knowledge such applications do not feature additional privacy enhancements to address concerns relating to the GDPR for Smart Cities or to mitigate the shortcomings of FL outlined above.

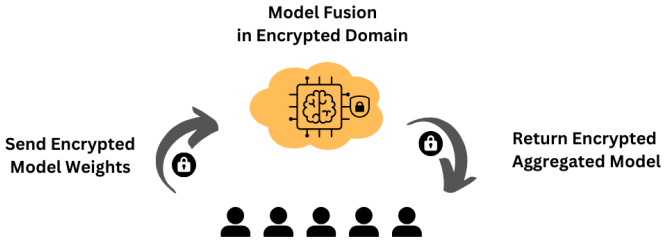


Fig. 1: Workflow of FL using HE. First the clients train their local models and encrypt the model weights. The encrypted weights are then sent to the server and aggregated to a global model without decrypting them. After that, the global model parameters are sent back to the clients. The clients decrypt the global parameters and use them to update their local models. This process is repeated until the global model converges or for a fixed number of rounds.

III. Fully Homomorphic Encryption

Homomorphic encryption is a cryptographic technique which allows for computations to be performed on encrypted data without the need to decrypt it. In the context of FL it can be used to encrypt the local model parameters of a client before sending it to the server to aggregate them with the parameters of the other clients participating in the training process. The aggregation will then be performed on the encrypted data in the form of computing a weighted average of the clients' model parameters. The weight of an individual clients is given by the size of its training dataset divided by the number of training samples across all clients. Afterwards, the still encrypted global model parameters are sent back to the clients where they can be decrypted. The local model is subsequently updated with the global parameters. By avoiding disclosure of the local plaintext models to the server it would effectively eliminate the possibility of attacks by the server on local models. The basic workflow of adding HE to FL is depicted in Figure 1.

Popular Fully Homomorphic Encryption (FHE) schemes are the CKKS scheme [8] and the BFV scheme [5]. The BFV scheme is an exact encryption scheme that supports modular arithmetic over finite fields. A major limitation of this scheme is that the message space is over integers. Hence, it cannot be applied in practical settings where input values might not be representable using only integers. CKKS is an approximate, asymmetric encryption scheme that supports arithmetic over complex numbers. The scheme incurs an approximation error after decryption for enabling computations over arbitrary floating-point numbers. The error introduced by the encryption will be kept within a small range based on chosen encryption parameters, so that it will not significantly influence the computation results [8]. Therefore, the parameters need to be set according to the needed accuracy for the computation task at hand. Despite the approximate nature of computations, CKKS provides more flexibility than BFV regarding its usability in practice due to its arbitrary input space. Hence, we will focus on the CKKS scheme in this paper.

The CKKS scheme has been implemented in several open-source libraries. The library that was used for the purposes of this paper is TenSEAL [3].

A. Threat Model

There are several possibilities in the process of FL where information about the clients could be obtained by parties not privy to that information. These scenarios include both malicious actors, as well as curious participants of the training process. These include:

- Honest-but-curious aggregator: The aggregator (server) follows the FL protocol without modifying the received data or the model sent back to the clients but still tries to extract as much information about the clients as possible.
- Collusion: Participants of the training process may choose to collude in order to send false data to disrupt the training or try to gain information of other participants.
- Outsider attacks: Attackers outside of the network may try to obtain information about the data on the edge devices by interacting with the actors involved in the FL process, either during or after training.

Ideally, all of these threat scenarios should be mitigated for FL to be secure. We assume that the communication between server and clients as well as the data storage of the clients is secure since this can be achieved through established cryptographic means and is not a problem unique to FL. HE can address some, but not all of these threat scenarios. Threats from an Honest-but-Curious aggregator are prevented by concealing the plaintext model parameters from it. However, the protection of HE is only active during training, hence, information leakage can still occur when postprocessing a model that was trained using FL even with the use of HE. This needs to be taken into account and mitigated by other means if possible.

Moreover, HE does not eliminate collusion attacks from multiple collaborating clients. This can be mitigated by utilizing multi-key approaches, such as the ones mentioned in Section IV.

IV. Related Work

Combining HE and FL has already been successfully utilized in the health domain. HE was incorporated into a FL COVID-19 detection algorithm using the BFV scheme in [34]. The algorithm is a binary classifier that detects COVID-19 infections based on the X-ray image of a patient's lung. The accuracy of the algorithm with HE was comparable to the unencrypted baseline and in some instances even better while the runtime of the HE approach increased significantly, especially when increasing the number of clients.

The use of HE was tested for IoT communications on the N-BaIoT dataset used for the detection of botnet attacks on IoT devices and achieved high accuracy, precision and recall in detecting malicious activity [15].

Additionally, HE was used in conjunction with FL to analyze network traffic [28]. Here, HE was incorporated into a model for predicting network traffic volume. This approach also yielded good results. Surprisingly, the HE implementation produced better results than the base FL approach in some instances. One possible explanation of this improvement is better generalization due to encryption noise acting as a regularization factor.

One shortcoming of the basic HE setup is that a shared secret key is used by all clients. Otherwise, the clients' decryption of the updated global model would be invalid. However, this leads to the possibility of attacks by adversarial clients. For example, a client could collude with the server to obtain the models of other clients to obtain sensitive information. This means that the basic HE approach assumes that the clients are trustworthy. To mitigate this problem, multi-key HE schemes have been proposed [7], [21]. Further, in [27], the authors propose a multi-key HE scheme that mitigates model poisoning attacks. Thus, shortcomings of plain HE outlined in Section III-A are improved upon. An approach for improving the performance of HE in a FL setting is given in [17]. Here, the authors utilize HE to encrypt model parameters only selectively. The ratio of encrypted parameters can be set depending on the need to decrease overhead introduced by the encryption. The authors were successful in reducing overhead and empirically demonstrated defense effectiveness against gradient inversion in [38] and language model inversion in [11]. In [32], a similar approach to ours was tested. Here, the authors implemented their own version of a multi-key approach based on [21] and, similar to our approach, integrated it into Flower, an open-source framework for FL [4]. The authors tested their approach on X-ray images for the COVID-19 detection use case of [34] mentioned above. The use of HE leads to a small drop in accuracy which mirrors our results. Notable for the practical application of HE is the potential for side-channel attacks on the CKKS on edge devices. In [2], the authors showcase the feasibility of such attacks against the CKKS implementation of the SEAL library. It is possible to deduce the secret key from power traces taken during the key

generation process using machine learning. This leads us to be cautious about the use of SEAL in practical settings in its current version.

V. FLCrypt

FLCrypt is the FL framework, which we used for the experiments described in this paper. It is largely based on open-source libraries. We used a custom version of Flower [4] for the FL setup and integrated the CKKS functionality of the TenSEAL library to encrypt client model updates before transmitting them to the server for aggregation. Moreover, we utilized the Hydra framework [35] to manage configurations for our experiments. For our customized Flower version, we added an extra payload field to the instructions and response classes for the client and server to make it easier to send the model parameters back and forth. For the encryption of model parameters, we utilize the CKKSVector class of TenSEAL. For this, the model parameters have to be flattened, meaning that the original shape has to be restored for the model update. This is done to decrease the size of the encrypted model. The FL functionality is provided by Flower. The only further changes to the framework are adjustments to the computations of the server-side aggregation in order to handle the encrypted values.

A. Experiments

The experiments were all conducted using FLCrypt on the metal ball data set published in [13]. The data set contains audio recordings of metal balls rolling down a steel slide as a part of a bigger track. It comprises three classes that correspond to different surface coatings of the metal balls, one of which is scratched. The audio was recorded using a low-cost microphone and the steel slide was surrounded with a casing to dampen background noise. This data set was originally created to improve machine learning applications for industrial sound analysis as there is a scarcity of usable data in that field. In this context the data set can be used to develop industrial acoustic quality control applications based on material conditions with emphasis on fault detection. According to the original paper, the dataset is relatively easy for the given classification task. A Deep Neural Network (DNN) baseline accuracy close to 99% was reported by the authors. Due to its simplicity, it provides a realistic initial target for IoT applications with limited computation power of the edge devices. We use it as a first step in showcasing the effectiveness of applying HE to audio event detection with the possibility of extending our approach to more complex use cases in the future.

For our FL experiments, we split the original balanced data set into three separate client partitions, consisting of 450 training and 57 test samples each. We ensure that each partition comprises a balanced number of examples corresponding to only two of the three classes, with the missing one different for each partition. The test sets were all evenly split into 19 samples from each of the three classes.

B. Setup

We use a HE implementation with a circuit length of two, a 128-bit security level and a precision of 2^{40} bits. To assess the effect of the HE we run two encrypted FL setups - FedAvg and FedProx - and compare them with the unencrypted baseline of FedAvg. Additionally, we also compare the strategies against each other.

We executed 20 rounds of FL using three clients. Each client locally trained for three epochs per round. To evaluate the aggregated global model, we measured the accuracy and loss on the local test data of each client. The model used by the clients is a Convolutional Neural Network (CNN) with two 2D convolutional layers and ReLU activation. The model also includes a batch normalization layer and an average pooling layer after each convolutional layer. The model has a total of 2745 parameters. The hyperparameters were the same across all clients and both strategies and are summarized in Table I.

TABLE I: The training hyperparameters used for the experiments.

Hyperparameter	Value
Learning rate	0.001
Batch size	64
Validation split	0.1
Optimizer	Adam
Loss	Softmax cross entropy

For the FedProx strategy a proximal parameter of $\mu = 0.02$ was used. The training hyperparameters were the same across all clients and both strategies and summarized in Table I.

To evaluate the runtime performance, we compare execution times of each FL process step. We also analyze transmission overheads introduced by HE by examining different sizes of serialized parameters before and after encryption. The hardware specifications that were used for the experiments are shown in Section V-B.

C. Evaluation

The FL runs with HE consistently show a similar accuracy as the unencrypted baseline. The FedAvg strategy as well FedProx get high accuracies of over 99% on the local client test sets. The average training loss and accuracy across three runs can be seen in Figure 2a. The graphs also show that the client models quickly converge to achieve near perfect accuracy. Compared to each other, the final accuracies of FedAvg and FedProx are the same but the client models converges faster for FedProx than for FedAvg as shown in Figure 2b. This suggests that training time can be saved by employing the FedProx strategy. Overall, the unencrypted FedAvg baseline model routinely achieves an accuracy of 1 shown in Figure 2c. Therefore, the drop in performance when using HE is marginal. This is in line with previous applications of HE to FL in other use cases of medical

TABLE II: The specification that were used for the experiments.

Spec	Server	Client
CPU	Intel Xeon CPU E5-2620 v4	Intel Xeon CPU E5-2650 v4
Graphics card	GeForce GTX 1080 Ti	GeForce GTX 1080 Ti
Memory	64 GB	128 GB

TABLE III: Comparison of runtime for different processes between encrypted and unencrypted runs.

Process	Time unencrypted (sec)	Time encrypted (sec)
Encryption	-	0.31
Decryption	-	0.031
Serialization	-	0.07
Deserialization	-	0.02
Server aggregation	0.00066	0.07
Complete run	27	48

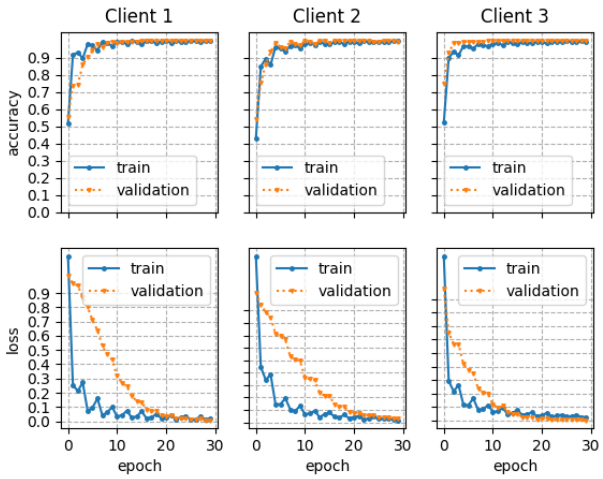
image classification [34] and network traffic prediction [28], described in Section IV. The approach in [15] also achieves an accuracy of over 99% for identifying malicious traffic in an IoT network. This indicates a general viability of using HE in the context of FL independent from the specific use case. Furthermore, [34] used an exact HE scheme which does not showcase a notable improvement in terms of accuracy over our use of the CKKS scheme. This indicates that the approximate nature of CKKS does not hurt model performance in practical applications while offering more flexibility regarding its use.

The difference in runtime performance between the HE and the unencrypted setup is significant. The baseline total execution time is 27 seconds on average across three runs of FL. After incorporating HE into the training, the overall runtime increases to around 48 seconds for both strategies. A more detailed runtime breakdown is provided in Table III. For the individual processes, the values are given as the average execution time of that process across 20 rounds of FL training. This includes all clients and the server. Plaintext serialization and deserialization in the baseline setup are carried out by the Flower library internally and are assumed to be negligible in our setup.

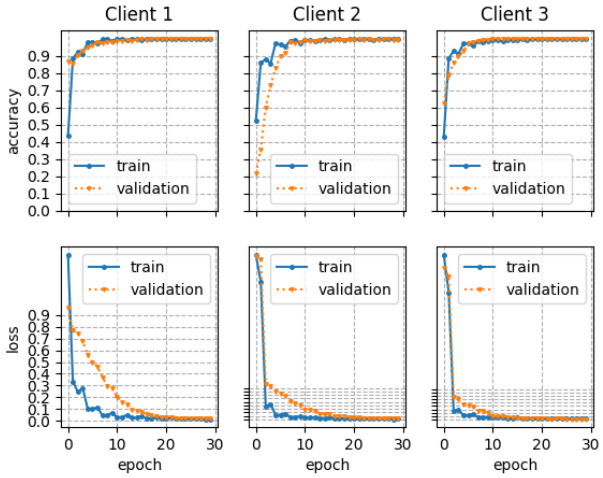
It can be observed that the runtime increase due to the encryption is not symmetric. The encryption and serialization processes take significantly longer than their counterparts. There is also a large increase in the server's aggregation time. In general, it can be observed that the encryption time is the biggest factor in the overall time increase on the client side.

In [34], the run time for 128-bit security and three clients was around 5000 seconds using the BFV scheme of SEAL which is significantly higher than our result. However, since the size of the model was not specified, it is unclear how much of that difference is due to the different schemes. In [28], the authors found a runtime increase from around 0.07 seconds for FL with plaintext to around 15.5 seconds when using the CKKS implementation in TenSEAL for a model with two layers with 400 neurons respectively. Unfortunately, the number of clients is not specified. Still, compared to our model with 2745 parameters this roughly scales with respect to the number of model parameters.

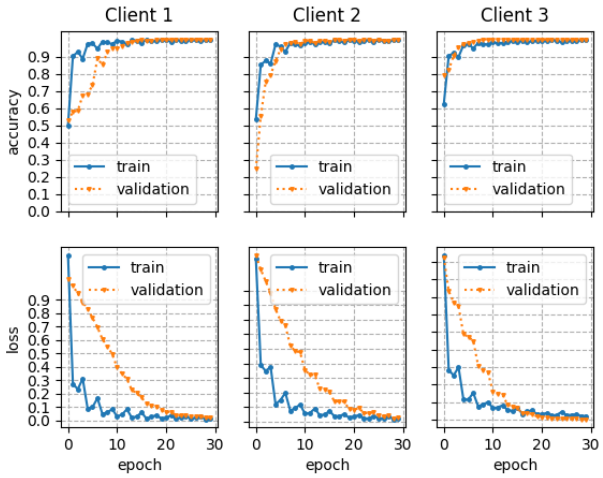
To analyze the development of the encrypted model size, we examine a randomly initialized $(768, 3)$ array of model parameters which corresponds to the largest layer of our model with entries ranging from zero to one. Unencrypted, it has a size of 18.6 KB. After encryption and serialization, its size grows to 301 KB. This marks an increase compared to the



(a) Accuracy and loss of the global model for the three clients using HE for FedAvg.



(b) Accuracy and loss of the global model for the three clients using HE for FedProx.



(c) Baseline accuracy and loss for FedAvg.

Fig. 2: Comparison of accuracy for the FL strategies with HE to the unencrypted baseline.

TABLE IV: Data size in relation to number of parameters.

Input array	Size original (KB)	Size encrypted (KB)	Number of parameters	Ratio
$(384, 3)$	9.3	301	1152	32.3
$(768, 3)$	18.6	301	2304	16.2
$(1536, 3)$	37	601	4608	16.2
$(3072, 3)$	74	902	9216	12.2
$(6144, 3)$	148	1504	18432	10.2
$(12288, 3)$	295	2708	36864	9.2
$(24576, 3)$	590	5414	73728	9.2
$(49152, 3)$	1180	10828	147456	9.2
$(98304, 3)$	2359	21658	294912	9.2

TABLE V: Projected size of well-known neural networks.

Input array	Size original (MB)	Size encrypted (MB)	Number of parameters (Million)
ResNet50	98	901	25.6
ResNet101	171	1573	44.7
ResNet152	232	2134	60.4
VGG16	528	4858	138.4
VGG19	549	5051	143.7
ConvNeXtSmall	192	1766	50.2
ConvNeXtBase	339	3110	88.5
ConvNeXtLarge	755	6946	197.7
MobileNet	16	147	4.3
NASNetMobile	23	212	5.3
EfficientNetB0	29	267	5.3
EfficientNetB7	256	2355	66.7

original array size by a factor of 32.2. For larger number of model parameters, the size increase settles on a ratio of 9.2. This matches the linear runtime increase in relation to the number of model parameters found in [25]. The encryption increases the size of the model by an order of magnitude. In total, we obtain an encrypted size of 4.2 MB for our model compared to an original size of 12.6 KB. This is because every encrypted model weight has a size of at least 0.3 MB, leading to a disproportionate increase in size for small models.

An overview of the ciphertext size for different numbers of parameters can be found in Table IV. From this, we also extrapolate the encrypted size of a selection of well-known neural networks shown in Table V.

D. Discussion

Our application of FHE to FL yields promising results for our use case of audio event detection. Although there is a small decrease in accuracy, the quality of the global model remains high while enhancing privacy. Still, the model performance under realistic environmental conditions warrants further investigation since HE could have a more detrimental effect in such settings. Further, it is still unclear how well our current approach alone would extent to settings with large models and a lot of clients. More research on that is needed. This is especially relevant for edge devices where computational power and memory capacity are limited. Still, this is promising first step. It should be noted that, to the best of our knowledge, no counter measure to the side-channel attack in [2] has been implemented in SEAL (release 4.1). Therefore, current real-world implementations of HE that use SEAL should be viewed with caution. Moreover, other libraries should be tested for similar attacks. Moreover, it should be considered that the choice of encryption parameters has a significant impact on the data size of the encrypted model. The results of this paper were all obtained for 128-bit security, 40-bit precision and a circuit length of two. Increasing these parameters leads to an increase of the data size. Notably, an increase in data size would also increase the computational overhead which could lead to problems in realistic deployments on energy efficient sensor nodes. Our results suggest that the strategy FedProx is preferable over FedAvg in such circumstances since model convergence can be achieved with less training time while achieving the same level of accuracy. Still, a tradeoff between security and computation power might be necessary in practice. However, a certain level of overhead needs to be accommodated to ensure the security of the encryption. Thus, choosing an appropriate set of encryption parameters should also take this into consideration. However, we have not yet analyzed the exact impact of adjusting these parameters in a rigorous manner. This would be an interesting question for further analysis, especially since more sophisticated FL strategies are likely to need more complex encryption parameters. Another possible avenue of further research is to improve upon the security of HE by integrating Differential Privacy into the current approach. Differential Privacy involves adding random noise to the output of an algorithm to masks the contribution of individual data points [9], thus, mitigating inference attacks on the model.

VI. Conclusion

We have successfully incorporated HE into FL for audio event classification for two different aggregation strategies and showed the effectiveness of our approach. Additionally, we have shown that FedProx retains advantages over FedAvg under HE. We have identified key reasons for a substantial runtime increase when using HE for FL that impede its scalability. We identified the number of model parameters as the main factor that determines the size of the encrypted data and analyzed how the encryption time develops in relation to it. This already showcases the utility of HE in certain settings, especially in our use case of sound classification where model size is moderate. While our experiments lead us to be optimistic, more work needs to be done analyze and improve the viability for HE in other practical settings since the linear increase relative to the number of model parameters might become infeasible for large model sizes, especially due to the rise of large-scale machine learning and the increasing prevalence of edge devices. Furthermore, the possibility of side-channel attacks needs to be further investigated and prevented before deployment in real scenarios.

Acknowledgements: This paper was made within the context of the SEC-Learn project on secure FL for edge devices at Fraunhofer IDMT. We would like to extend our gratitude to our colleagues for the fruitful discussions about the results that were obtained and to Sebastian Speitel for his prototype of FLCrypt.

References

[1] Alías, F., Alsina-Pagès, R.M.: Review of wireless acoustic sensor networks for environmental noise monitoring in

smart cities. Journal of Sensors **2019**(1), 7634860 (2019). https://doi.org/https://doi.org/10.1155/2019/7634860
[2] Aydin, F., Aysu, A.: Leaking secrets in homomorphic encryption with side-channel attacks. Journal of Cryptographic Engineering (2024). https://doi.org/10.1007/s13389-023-00340-2
[3] Benaissa, A., et al.: Tenseal: A library for encrypted tensor operations using homomorphic encryption (2021)
[4] Beutel, D.J., et al.: Flower: A friendly federated learning research framework (2022)
[5] Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-lwe and security for key dependent messages. In: Annual cryptology conference. pp. 505–524. Springer (2011)
[6] Carlini, N., et al.: Membership inference attacks from first principles. In: 2022 IEEE Symposium on Security and Privacy (SP). pp. 1897–1914. IEEE (2022)
[7] Chen, H., et al.: Multi-key homomorphic encryption from tfhe. In: Advances in Cryptology–ASIACRYPT 2019: 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8–12, 2019, Proceedings, Part II 25. pp. 446–472. Springer (2019)
[8] Cheon, J.H., et al.: 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. pp. 409–437. Springer (2017)
[9] Dwork, C., et al.: Calibrating noise to sensitivity in private data analysis. In: Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3. pp. 265–284. Springer (2006)
[10] Feraudo, A., et al.: Colearn: Enabling federated learning in mud-compliant iot edge networks. In: Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking. pp. 25–30 (2020)
[11] Fowl, L.H., et al.: Decepticons: Corrupted transformers breach privacy in federated learning for language models. In: The Eleventh International Conference on Learning Representations (2022)
[12] Fredrikson, M., et al.: Model inversion attacks that exploit confidence information and basic countermeasures. In: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security. pp. 1322–1333 (2015)
[13] Grollmisch, S., et al.: Sounding industry: Challenges and datasets for industrial sound analysis. In: 2019 27th European Signal Processing Conference (EUSIPCO). pp. 1–5. IEEE (2019)
[14] Gudur, G.K., Perepu, S.K.: Zero-Shot Federated Learning with New Classes for Audio Classification. In: Proc. Interspeech 2021. pp. 1579–1583 (2021). https://doi.org/10.21437/Interspeech.2021-2264
[15] Hijazi, N.M., et al.: Secure federated learning with fully homomorphic encryption for iot communications. IEEE Internet of Things Journal (2023)
[16] Horvitz, E., Mulligan, D.: Data, privacy, and the greater good. Science **349**(6245), 253–255 (2015)
[17] Jin, W., et al.: FedML-HE: An efficient homomorphic-encryption-based privacy-preserving federated learning system. In: International Workshop on Federated Learning in the Age of Foundation Models in Conjunction with NeurIPS 2023 (2023)
[18] Johnson, D.S., et al.: Desed-fl and urban-fl: Federated learning datasets for sound event detection. In: 2021 29th European Signal Processing Conference (EUSIPCO). pp. 556–560. IEEE (2021)
[19] Li, T., et al.: Federated learning: Challenges, methods, and future directions. IEEE signal processing magazine **37**(3), 50–60 (2020)
[20] Li, T., et al.: Federated optimization in heterogeneous networks. In: Dhillon, I., Papailiopoulos, D., Sze, V. (eds.) Proceedings of Machine Learning and Systems. vol. 2, pp. 429–450 (2020)
[21] Ma, J., et al.: Privacy-preserving federated learning based on multi-key homomorphic encryption. International Journal of Intelligent Systems **37**(9), 5880–5901 (2022)
[22] Ma, X., et al.: A state-of-the-art survey on solving non-iid data in federated learning. Future Gener. Comput. Syst. **135**(C), 244–258 (oct 2022). https://doi.org/10.1016/j.future.2022.05.003
[23] McMahan, B., et al.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017)
[24] Ntalampiras, S.: Moving vehicle classification using wireless acoustic sensor networks. IEEE Transactions on Emerging Topics in Computational Intelligence **2**(2), 129–138 (2018). https://doi.org/10.1109/TETCI.2017.2783340
[25] Park, J., Lim, H.: Privacy-preserving federated learning using homomorphic encryption. Applied Sciences **12**(2) (2022). https://doi.org/10.3390/app12020734
[26] Peng, H., et al.: Fedef: Federated learning for heterogeneous and class imbalance data. In: 2023 IEEE Symposium on Computers and Communications (ISCC). pp. 619–624. IEEE (2023)
[27] Rahulamathavan, Y., et al.: Fhefl: Fully homomorphic encryption friendly privacy-preserving federated learning with byzantine users (2023)
[28] Sanon, S.P., et al.: Secure federated learning: An evaluation of homomorphic encrypted network traffic prediction. In: 2023 IEEE 20th Consumer Communications & Networking Conference (CCNC). pp. 1–6. IEEE (2023)
[29] Sattler, F., et al.: Robust and communication-efficient federated learning from non-iid data. IEEE transactions on neural networks and learning systems **31**(9), 3400–3413 (2019)
[30] Shokri, R., et al.: Membership inference attacks against machine learning models. In: 2017 IEEE symposium on security and privacy (SP). pp. 3–18. IEEE (2017)
[31] Turchet, L., et al.: The internet of sounds: Convergent trends, insights, and future directions. IEEE Internet of Things Journal **10**(13), 11264–11292 (2023). https://doi.org/10.1109/JIOT.2023.3253602
[32] Walskaar, I., et al.: A practical implementation of medical privacy-preserving federated learning using multi-key homomorphic encryption and flower framework. Cryptography **7**(4) (2023). https://doi.org/10.3390/cryptography7040048
[33] Wang, K., et al.: Vehicle recognition in acoustic sensor networks via sparse representation. In: 2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW). pp. 1–4 (2014). https://doi.org/10.1109/ICMEW.2014.6890549
[34] Wibawa, F., et al.: Homomorphic encryption and federated learning based privacy-preserving cnn training: Covid-19 detection use-case. In: Proceedings of the 2022 European Interdisciplinary Cybersecurity Conference. pp. 85–90 (2022)
[35] Yadan, O.: Hydra - a framework for elegantly configuring complex applications. Github (2019)
[36] Ye, J., et al.: Enhanced membership inference attacks against machine learning models. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. pp. 3093–3106 (2022)
[37] Zhang, Z., et al.: Semi-supervised federated learning with non-iid data: Algorithm and system design. In: 2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys). pp. 157–164. IEEE (2021)
[38] Zhu, L., et al.: Deep leakage from gradients. Advances in neural information processing systems **32** (2019)