Privacy-Preserving Keystroke Analysis using Fully Homomorphic Encryption & Differential Privacy

Jatan Loya [1] **Tejas Bana** [2]

Abstract

Keystroke dynamics is a behavioural biometric form of authentication based on the inherent typing behaviour of an individual. While this technique is gaining traction, protecting the privacy of the users is of utmost importance. Fully Homomorphic Encryption is a technique that allows performing computation on encrypted data, which enables processing of sensitive data in an untrusted environment. FHE is also known to be "future-proof" since it is a lattice-based cryptosystem that is regarded as quantum-safe. It has seen significant performance improvements over the years with substantially increased developer-friendly tools. We propose a neural network for keystroke analysis trained using differential privacy to speed up training while preserving privacy and predicting on encrypted data using FHE to keep the users' privacy intact while offering sufficient usability.

1. Introduction

Fully Homomorphic Encryption (FHE) (Gentry, 2009) is often considered as the "holy grail" of encryption since it enables computation on encrypted data. Traditionally, data is encrypted at rest and in transit. However, by allowing computation on encrypted data, FHE allows for delegation of computation for third-party and secure multi-party computation (MPC) (Goldreich, 1998). It eliminates the need for a tradeoff between data usability and data privacy since there is no need to anonymize or drop some data features.

With the recent soar in machine learning as a service (MLaaS), a large volume of user data is being sent to prediction models on the cloud. The users' request is processed in an unencrypted form on the cloud, leading to data collection and exposure due to data breach. Sometimes, sensitive information like healthcare and financial data cannot be transferred to a third party for processing due to compliance and regulations. By leveraging FHE, a user can encrypt their data with FHE and send it to a server. The server performs the required computation over encrypted data and sends back the encrypted result that only the user can decrypt. This secure delegation of computation expands the horizon by providing services requiring sharing sensitive data while preserving data privacy. Such applications range from DNA analysis, sharing medical data, insurance data, financial analysis and even location data analysis. Until recent times, training a machine learning model with FHE has been a tedious task. There have been recent advances, like Cryptonets (Gilad-Bachrach et al., 2016)which achieves an accuracy of 99% on MNIST character recognition dataset(LeCun, 1998).

We propose a neural network for user authentication using FHE and Differential privacy (Dwork et al., 2006)(Dwork et al., 2014). While training a neural network by encrypting the model's parameters and the training data seems plausible, it requires too much computation and time for a simple multi-classification problem. Hence, we propose a method in which neural networks are trained using differential privacy then evaluated on homomorphically encrypted data. By doing so, we can leverage the benefits of both FHE and differential privacy while maintaining the time complexity of the training.

2. Background

2.1. Fully Homomorphic Encryption (FHE)

It is a form of encryption that allows users to perform computation on encrypted data without decrypting it first. The computation result is encrypted, and when decrypted, is identical to the output produced when performing the same computation on plain text.

In the scheme proposed by Cheon et al. (Cheon et al., 2017), which we will refer as CKKS is an HE scheme for performing arithmetic operations of approximate numbers. This scheme allows us to add noise following the significant bits which contain the main message. In the

[*]Equal contribution [1]Department of Computer Engineering, Vishwakarma Institute of Technology, Pune, India [2]Department of Information Technology, D.Y Patil College of Engineering, Pune, India. Correspondence to: Jatan Loya <jatan.loya18@vit.edu>, Tejas Bana <tejasbana@gmail.com>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

CKKS scheme, the encryption noise is considered part of the error while performing approximate computation. According to the authors, encryption c of message m by the secret keys k will have a decryption structure of the form $< c, sk >= m + e(modq)$ where e is a small error inserted to guarantee the security of hardness assumptions such as the learning with errors (LWE) and ring-LWE (RLWE) as described in the CKKS paper (Cheon et al., 2017). If the size of e is negligible compared to the message, then most likely, it will not distort significant figures of m, and the total value $m' = m + e$ is used to replace the original message. A scaling factor can be multiplied with the message before encryption to reduce precision loss from encryption noise. This results in an approximate value of plaintext with predetermined precision. For every multiplication performed, some noise is added in the ciphertext, which can also be reduced by rescaling the ciphertext.

In our experiments, we have used TenSEAL (Benaissa et al., 2021) which is an open-source library that enables computation on encrypted tensor, which can be converted from machine learning frameworks like Pytorch and Tensorflow written in Python, which wraps Microsoft SEAL (Simple Encrypted Arithmetic Library) (Chen et al., 2017).

2.2. Differential Privacy

Differential Privacy is a technique for training a machine learning model by limiting the data leakage about specific data points in the training data. The tradeoff of using differential privacy is loss in model accuracy.

As mentioned by authors (Bagdasaryan et al., 2019), using differentially private stochastic gradient descent (DP-SGD) (Rajkumar & Agarwal, 2012) causes more accuracy drop for underrepresented classes and subgroups. They have also uncovered that DP-SGD amplifies the model's "bias" towards the most popular elements of distribution. For implementing differential privacy, we have used an open-source library Opacus developed by PyTorch.

2.3. Keystroke Dynamics

Keystroke Dynamics is a form of biometric authentication based on inherent typing behaviour which differs from person to person. This is aimed to deter malicious actors from using a compromised password for gaining authentication. Using keystroke dynamics, such malicious attempts can be detected and denied since their typing rhythm differs from a genuine user. Various anomaly-detection algorithms have been proposed which can be used for detecting an intruder who is entering a compromised password.

In the research conducted by Killourhy et al. (Killourhy & Maxion, 2009), the authors have collected a keystroke dynamics dataset and have proposed a repeatable evaluation procedure. The data was collected from 51 people typing 400 passwords each. The password was ".tie5Roanl", which is a strong password according to the publicly available password generator they have used. A set of timings features were extracted from the raw typing data. The Enter key was considered part of the password since it can mask a 9-character password ten keystrokes long. Also, keyup-keydown times, keydown-keydown times and hold times for all keys were extracted. 31 timings features were extracted and put into a vector of floating-point numbers.

3. Proposed Method

We have used the CKKS scheme, which is a levelled homomorphic encryption scheme. Depending on the parameters selected for encryption, a limit is set on the number of multiplication operations that can be performed on the encrypted data, which directly impacts the depth of the machine learning model that could be used. Machine learning models use non-linear activation functions, which are needed to be approximated using polynomials for testing on data encrypted using the CKKS scheme.

Generally, ReLU (Rectified Linear Units) is preferred for model training since it helps converge the training loss faster. Since ReLU is mathematically defined as $y = max(0, x)$. We implement a ReLU function using a polynomial approximation. In the research by Gottemukkula (Gottemukkula, 2019), the authors proposed several polynomial approximations for the ReLU function. Since our objectives include minimising the computational depth, the ReLU approximation of polynomial of degree 2 was used to test the accuracy. $f(x) = 0.47 + 0.50x + 0.09x^2$, $x \in [-\sqrt{2}, \sqrt{2}]$.

For ReLU approximation to work, inputs to the activation should lie between $-\sqrt{2}$ and $\sqrt{2}$, for which data was normalised between $[0 - 1]$, and training included weight decay.

For using sigmoid as an activation function the equation was approximated to the polynomial function 0.5 + 0.197 * x - 0.004 * x^3. Due to the cubic factor in sigmoid approximation, there is a risk of surpassing the multiplication limit of the scheme. Since this sigmoid approximation is only good in the range $[-5, 5]$, models should have all the inputs in that range. In order to do this, the models' parameters should be as small as possible. This can be done by apply regularisation like weight decay which resulted in reducing the final accuracy as shown in Table 1. The number of hidden layers had to be increased for this model to perform better under the effects of differential privacy and regularisation techniques.

Since we could only limit our model depth to 2 layers considering every multiplication in FHE involves approximation or adds fuzzy noise, increasing the number of hidden layers further resulted in all prediction of FHE encrypted data to belong to the same class. We tested this for the CKKS Poly Mod Degree 8192 and 16384 since it was within the multiplicative depth bound. Also, the time taken to evaluate encrypted data is directly proportional to the number of layers and activation function used. All these factors were considered while choosing the activation function.

Table 1. Comparison of Activation Function using Differential Privacy

Activation Function	Accuracy
ReLU	91.13%
ReLU Approximation	78.28%
Sigmoid Approximation	74.50%
X2	84.80%

For this reason, we have used x^2 as the activation function. Usually, x^2 makes the training unstable compared to other activation functions like ReLU, which slightly decreases the accuracy as shown in Figure 2. Since we could not use ReLU, we opted for x^2 as the activation function between hidden layers.

Higher approximation while performing multiplication with ciphertext induces an unnecessary biased, resulting in the wrong prediction when dealing with classification containing many classes since predicted confidence values of classes for multi-class predictions are very close to each other.

For solving the above issue, we found that high dropout probability between the hidden layers while training yielded significantly better results as the confidence of the predicted class was much higher than confidence for other classes during evaluation.

3.1. Model

The proposed neural network model has two layers with 1536 neurons in the first layer since adding more neurons to it did not improve the result. And 51 output neurons for 51 classes. The batch size is 64. In Figure 1, i represents the input attributes (31), k represent output classes (51), n represents no. of neurons in the first layer (1536).

4. Experiment And Results

The model was trained for 900 epochs with a batch size equal to 64 and an initial learning rate of $2e^{-4}$. We reduced the learning rate by a factor of 10 after every 300 epochs of training. For Loss calculation, we used the Cross-Entropy loss function. After the first linear layer, dropout with a probability of 80% was used. For the non-linear activation function, we used x^2. Adam optimiser was used because it converges much faster and provides lower loss values for the

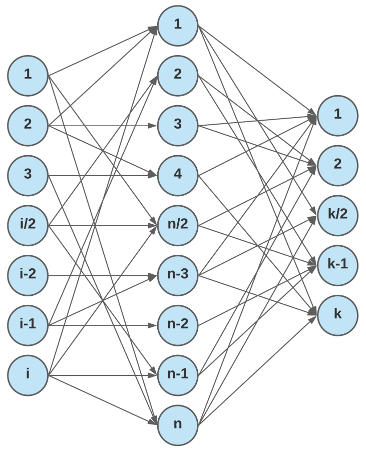

Figure 1. Neural network architecture

model, which empirically was favourable for this training compared to other optimisers.

For differential privacy, the following parameters in Opacus library were used: $sample\ rate$ = $1e^{-3}$, $noise\ multiplier$ = $2e^{-1}$, $per\ sample\ gradient\ clipping$ = 1, $delta$ (probability of information getting accidentally leaked) = $1e^{-5}$. We considered two parameters for our FHE configuration, Param 1 and Param 2, as given in Table 2. Table 3 includes time taken by each layer of the model for different activation functions and FHE parameters. Since Param 1 is smaller than Param 2, Param 1 computations are up to $2.9\times$ faster than Param 2 computations except after prediction for the sigmoid activation function. A significant amount of time is required by the output layer for both activation function and parameter settings.

Table 2. Param & Poly Mod Degree & Coeff mod & Global scale

Parameter	Poly Mod Degree	Coeff mod	Global scale
Param1	8192	(31,26,26,26,26,26,26,31)	2^{26}
Param2	16384	(60,40,40,40,40,40,40,60)	2^{40}

Table 3. Time taken by each layer in seconds

Time taken	Sigmoid		X2	
	P1	P2	P1	P2
Data Encryption	0.0098	0.0232	0.0109	0.0232
First Layer	0.3669	0.9538	0.4015	0.9560
Activation Layer	0.0316	0.0817	0.0119	0.0282
Output Layer	13.8027	40.0593	18.0093	51.6791
Data decryption	0.0025	0.0059	0.0033	0.0074
After prediction	0.0016	0.0014	0.0017	0.0020
Total	14.2151	41.1253	18.4386	52.6959

Suppose Fully Homomorphic Encryption parameters are chosen carefully, the accuracy drop while evaluation is negligible compared with plaintext evaluation. The difference of accuracies between plaintext and encrypted evaluation are 0.2451 and 0.9804 for each parameter, respectively. The more critical factor for maintaining the accuracy is the combination of Differential Privacy parameters, activation function compatible with FHE evaluation, data normalization and the regularization techniques used for training the model.

Table 4. Accuracy of model

Training method	Param 1		Param 2	
	Plaintext	FHE	Plaintext	FHE
Differential Privacy	84.8039%	84.5588%	84.8039%	83.8235%
Clean data	86.2745%	85.4901%	86.2745%	–

5. Limitations & Conclusion

Using the CKKS scheme limits the multiplication depth and restricts us to only use approximations of non-linear activation functions. FHE model requires more computation resources and time, which might not be ideal for specific use cases where inference time is critical.

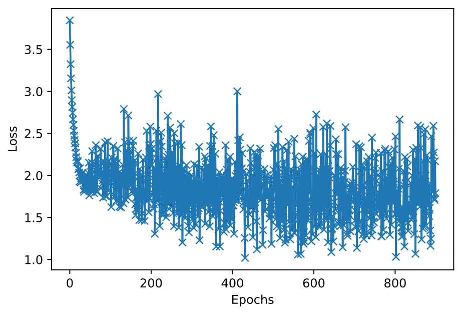

Figure 2. Loss vs Epoch for x^2

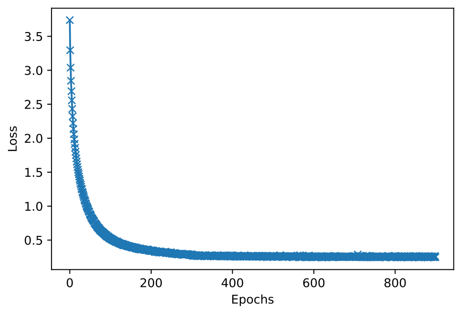

Figure 3. Loss vs Epoch for ReLU

Finding appropriate parameters for a problem that has a high number of classes is challenging and not straightforward. Figure 2 shows instability in training; hence using an approximation of ReLU with better fine-tuning of parameters will be our next step.

In conclusion, our work shows feasibility for developing privacy-preserving techniques for machine learning algorithms. Training using differential privacy offers better performance and can be used as a training alternative until FHE training becomes practical. Using encrypted data for inference from machine learning can enable a higher degree of security.

6. Future Work

Parameter selection for FHE is a crucial part that directly impacts performance and restricts the depth of multiplication. Selection of parameters requires manual trial-error, which is time-consuming and might not yield good result. More intelligent compilers like EVA (Dathathri et al., 2020) can be used for faster and more efficient parameter selection. Configuring FHE parameters and using techniques like rescaling, bootstrapping and RLWE to increase the number of hidden layers in the neural network to enable the use of larger models. Encrypting the model weights can also protect it from white-box adversarial attack.

References

Bagdasaryan, E., Poursaeed, O., and Shmatikov, V. Differential privacy has disparate impact on model accuracy. *Advances in Neural Information Processing Systems*, 32: 15479–15488, 2019.

Benaissa, A., Retiat, B., Cebere, B., and Belfedhal, A. E. Tenseal: A library for encrypted tensor operations using homomorphic encryption. *arXiv preprint arXiv:2104.03152*, 2021.

Chen, H., Laine, K., and Player, R. Simple encrypted arithmetic library-seal v2. 1. In *International Conference on Financial Cryptography and Data Security*, pp. 3–18. Springer, 2017.

Cheon, J. H., Kim, A., Kim, M., and Song, Y. Homomorphic encryption for arithmetic of approximate numbers. In *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 409–437. Springer, 2017.

Dathathri, R., Kostova, B., Saarikivi, O., Dai, W., Laine, K., and Musuvathi, M. Eva: An encrypted vector arithmetic language and compiler for efficient homomorphic computation. In *Proceedings of the 41st ACM SIGPLAN*

Conference on Programming Language Design and Implementation, pp. 546–561, 2020.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pp. 265–284. Springer, 2006.

Dwork, C., Roth, A., et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

Gentry, C. Fully homomorphic encryption using ideal lattices. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 169–178, 2009.

Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K., Naehrig, M., and Wernsing, J. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International conference on machine learning*, pp. 201–210. PMLR, 2016.

Goldreich, O. Secure multi-party computation (manuscript, preliminary version 78), 1998.

Gottemukkula, V. Polynomial activation functions. 2019.

Killourhy, K. S. and Maxion, R. A. Comparing anomaly-detection algorithms for keystroke dynamics. In *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, pp. 125–134. IEEE, 2009.

LeCun, Y. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Rajkumar, A. and Agarwal, S. A differentially private stochastic gradient descent algorithm for multiparty classification. In Lawrence, N. D. and Girolami, M. (eds.), *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pp. 933–941, La Palma, Canary Islands, 21–23 Apr 2012. PMLR. URL http://proceedings.mlr.press/v22/rajkumar12.html.