PRIVACY-PRESERVING PARALINGUISTIC TASKS

Francisco Teixeira, Alberto Abad, Isabel Trancoso *

INESC-ID / Instituto Superior Técnico, University of Lisbon, Portugal

ABSTRACT

Speech is one of the primary means of communication for humans. It can be viewed as a carrier for information on several levels as it conveys not only the meaning and intention predetermined by a speaker, but also paralinguistic and extralinguistic information about the speaker's age, gender, personality, emotional state, health state and affect. This makes it a particularly sensitive biometric, that should be protected. In this work we intent to explore how Leveled Homomorphic Encryption can be combined with a Neural Network to create a privacy-preserving machine learning framework for speech-based health-related tasks. In particular, we will apply this framework to the detection and assessment of a Cold, Depression and Parkinson's Disease. Moreover, we will show how using a Quantized Neural Network, with discretized weights, allows us to apply a Leveled Homomorphic Encryption technique called *batching* that can be utilized to reduce the effective computational cost of this framework.

Index Terms— Privacy, Machine Learning, Homomorphic Encryption, Speech, Health

1. INTRODUCTION

The widespread use of devices with internet access, together with the emerging market for data mining applications has raised concerns over the level of privacy currently given to users. Taking advantage of increasingly accurate Machine Learning algorithms, many Machine Learning as a Service providers use sensitive data to extract information and make predictions about the characteristics of their users.

Among other data types, speech stands out for the amount of information it holds. Aside from the linguistic content, from speech one can obtain other information, such as the speaker's age, gender, health and personality traits. However, the reasons that make speech useful also make it a target for malicious third parties intending to obtain sensitive information about unsuspecting users. This is especially true for health-related applications where a system may try to uncover whether someone presents symptoms of a medical condition, as this information is deeply sensitive.

*This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2019.

Privacy in speech is a fairly recent topic of research. One of the first strides in this direction was made by Pathak et al. [1], who adapted a Gaussian Mixture Model (GMM) to work with Homomorphic Encryption (HE), to perform both speaker verification and identification. In a different approach, Portêlo et al. [2], and later Jiménez et al. [3], applied Secure Binary Embeddings (SBE) and Secure Modular Hashing (SMH) to speaker verification. More recently Dias et al. [4] applied SMH and HE to an emotion recognition task, while Teixeira et al. [5] applied HE to several health-related paralinguistic tasks.

Outside speech, the literature is extensive, and a large amount of different techniques has been developed for Privacy-preserving Machine Learning. An important contribution to Privacy-preserving Machine Learning was recently proposed in Cryptonets [6]. In their work, the authors performed the prediction stage of a Neural Network, using Leveled Homomorphic Encryption (LHE), by replacing all operations with their homomorphic counterparts. This work was further improved by Chabanne et al. [7] and Hesamifard et al. [8]. This type of approach has the advantage of requiring only two rounds of communication between the client and the service provider, while maintaining privacy for the user's data at all times, as well as allowing the network's architecture to remain undisclosed. However, it has the disadvantage of entailing a large computational overhead. To avoid this overhead, other approaches have used Oblivious Transfer (OT) and Garbled Circuits (GC) instead, obtaining very good performance results, with low communication and computational costs [9][10][11][12]. Nevertheless, these approaches do not guarantee the model's privacy.

Some LHE schemes allow several values to be batched into a single ciphertext, which can be operated on as *SIMD (Single Instruction, Multiple Data)*, with a technique called *batching*. Since most operations on Neural Networks are vectorizable, *batching* allows several encrypted predictions to be made at the same time, making this technique particularly interesting for speech applications where predictions may be made at the frame level, utterance level, or even at speaker level. Although this technique had already been applied in Cryptonets, it was not used in our previous work [5]. For this reason, in this paper we describe how a Neural Network and its inputs have to be modified in order to allow the use of *batching* in an LHE context, and we provide a proof-of-

concept on how this modified network can be applied to three speech-based health-related tasks: the detection and assessment of a Cold, Depression and Parkinson's Disease.

We will not consider performing the training stage in a secure setting. In fact, due to the accuracy and computational losses imposed by secure frameworks, few works perform the training stage in a privacy setting, nevertheless, there are some notable exceptions [10][13].

The paper is organized as follows: Section 2 introduces HE. The approach implemented is introduced in Section 3. Section 4 describes the experimental setup as well as details on the implementation of the network. Section 5 includes the results obtained, together with a critical analysis. Section 6 presents our main conclusions and contributions.

2. HOMOMORPHIC ENCRYPTION

Homomorphic Encryption (HE) is a type of cryptosystem in which certain operations performed on *ciphertexts* (i.e. encrypted values) are *homomorphic* with regard to the *plaintexts* (i.e. unencrypted values). In other words, considering the encryption of a value x, $E(x)$ and of a value y, $E(y)$, if a homomorphic operation is performed on the two ciphertexts, the result of this operation will correspond to the equivalent unencrypted operation of the two values, as follows:

$$\begin{aligned} E(x) \otimes E(y) &= E(x \times y), \\ E(x) \oplus E(y) &= E(x + y), \end{aligned} \tag{1}$$

with $\otimes$ and $\oplus$ corresponding to homomorphic multiplication and addition, respectively.

Most HE schemes are limited either by the type and amount of operations that can be performed. This may be due to the construction of the scheme and due to its computational complexity. A particular type of HE is Leveled Homomorphic Encryption (LHE). LHE schemes allow the user to select the encryption parameters in such a way that it is possible to determine the maximum amount of operations that can be performed on the ciphertexts, while still being possible to decrypt them correctly. To perform more operations one has to increase the size of the encryption parameters, however, this means increasing the computational complexity of each operation in the scheme. To compensate for this computational limitation, Brakerski et al. [14] introduced *batching*, allowing several messages to be encrypted in the same ciphertext and thus, to be operated on at the same time.

3. PRIVACY-PRESERVING NEURAL NETWORKS

In order to adapt a Neural Network to HE, and transform it into an Encrypted Neural Network (ENN), it is first necessary to replace every operation by its HE counterpart. For Fully Connected (FC) layers, this is simply a matter of replacing additions and multiplications with their HE equivalent. However, nonlinear functions in the network (e.g. activation layers) cannot be computed, and need to be replaced with polynomials. To overcome this, in this work we applied the approach of Chabanne et al. [7] and Hesamifard et al. [8]. For this purpose, we used activation functions approximated with Chebyshev polynomials [8], taken from [5]. Since these have a limited approximation interval, we introduce a Batch Normalization (BN) layer before each Activation layer, to guarantee that its inputs fall within the convergence interval of the polynomial, as suggested by Chabanne et al. [7]. More concretely, we used Equation 2 as an approximation of the ReLU.

$$p(x) = 0.03664x^2 + 0.5x + 1.7056 \tag{2}$$

As stated in the Section 1, the use of *batching* may entail several advantages in terms of the performance of the ENNs, in particular for speech. However, its use is incompatible with other encoding techniques that allow fractional values to be encoded into LHE plaintexts. As such, *batching* requires the network's weights and inputs to be discretized. Following the approach of [6] and [8], in this work, this is done by scaling the weights up to a fixed precision. Nevertheless, this has to be done with care, since, in an encrypted context, if the values grow larger than a threshold value, they will yield incorrect results after decryption. Considering that the activation layers used in our networks are 2^{nd} degree polynomials, if a weight is multiplied by a scaling factor, when an input is forwarded through the activation layer, the factor will be squared. As such, propagating scaled values through the network will result in a very rapid growth of the scaling factor. Thus the scaling factor has to be selected considering the largest absolute value allowed by encryption scheme. The same problem holds when considering the inputs, but in this case we can avoid this problem by quantizing the input features instead, which allows us to feed them directly to the network as integers.

4. EXPERIMENTAL SETUP

To determine how feature quantization and weight discretization affects the performance of the ENN we performed experiments for a baseline ENN, for a network with quantized inputs (QNN), as well as for a network with scaled weights and quantized inputs (Scaled QNN). Since the Scaled QNN is the only network where every parameter and input is discretized, we could only use *batching* with it.

4.1. Datasets

Our experiments were conducted for three different speech affecting conditions, Cold, Depression and Parkinson's Disease. For Cold we used the Upper Respiratory Tract Infection Corpus (URTIC) [15] to perform a classification task. This corpus was used in Interspeech's 2017 ComParE Challenge [16]. The experiments concerning Depression were performed using the Distress Analysis Interview Corpus - Wizard

of Oz (DAIC-WOZ) [17], for both regression and classification, as in 2016's AVEC Challenge [18]. Finally for Parkinson's Disease (PD), the Parkinson's Disease Spanish Corpus was used for a regression task [19]. This corpus was provided for the 2015's Interspeech ComParE Challenge [20]. Since the labels corresponding to the test set were not available, all the results reported correspond to the development set of each corpus. A detailed description of each of the three datasets is provided in [15], [17], and [19].

4.2. Features

Two different feature sets were used in our experiments. For the experiments performed on the Depression and Cold datasets, the eGeMAPS feature set was used. However, for the Parkinson's Disease experiment, eGeMAPS did not achieve significant results. Consequently, a Parkinson's Disease specific feature set was used instead. Developed by Pompili et al. [21], this feature set contains 36 GeMAPS based features, along with 78 MFCC based features, resulting in a 114 dimensional feature vector. Both feature sets were extracted from audio files using openSMILE configuration files [22]. All features were zero-centered and normalized with unit variance using the mean and standard deviation computed from each training set.

4.3. μ-Law Quantization

As was stated in Section 3, to turn the input features into integers we decided to quantize them. To this end, we first used the μ-Law companding transformation, displayed in Equation 3, which maps inputs from $[-1, 1]$ to $[0, 1]$.

$$f(x) = sign(x)\frac{ln(1+\mu|x|)}{ln(1+\mu)} \tag{3}$$

Subsequently, to map the outputs of the function to $[0, \mu-1] \in Z$, where μ is the number of quantization channels, we used the quantization function in Equation 4.

$$Q(x) \;=\; \left\lfloor \mu\frac{f(x)+1}{2} + \frac{1}{2} \right\rfloor \tag{4}$$

Additionally, to promote smaller absolute values, each feature vector was zero-centered relative to its median value.

4.4. Encryption Parameters

In this work, we used Microsoft's Simple Encryption Arithmetic Library (SEAL) [23], which implements the LHE Fan and Vercauteren (FV) scheme. This library implements integer and fractional value encoders, which allow us to directly implement the network without any modification to the inputs and the network's weights. However, although the library also implements *batching*, these encoders are incompatible with its use. Some parameters of this library must be selected by the user: the plaintext modulus, the polynomial modulus, the coefficient modulus, the encoder's expansion base and its number of coefficients.

For the baseline ENN and for the QNN, we used a polynomial modulus of 8,192, a plaintext modulus of 2^{30}, and encoded weights and input features using SEAL's Fractional Encoder, with an expansion base of 3 and 16 coefficients for both the integer and the fractional part. For the Scaled QNN, the polynomial modulus was chosen to be 16,384. For batching to work, it is necessary that the plaintext modulus t be a prime number, congruent to $1 \; mod \; 2n$, with n being the polynomial modulus [23]. To account for the large values that arise from scaling the network's weights, we chose a plaintext modulus larger than 2^{59}, that fulfills the condition defined above. In all networks, SEAL's default value for a security level of 128 bits was selected for the coefficient modulus.

The networks were implemented in C++, using SEAL, and the encrypted predictions were computed using multithreading with 24 Intel(R) Xeon(R) CPU E5-2630 v2 @ 2.60GHz processors, in a machine with 250 GB of RAM.

4.5. Implementation

Considering that, in SEAL, the maximum value any underlying encrypted variable can take is 2^{60} [1], and that the scaling factor s grows with each layer, we have to ensure that there is only a minimum amount of parameters that need to be scaled. As such, in this section we describe how some operations can be pre-computed, in order to reduce the growth of the scaling factor. For the reasons stated in Section 3, between each pair of FC and Activation layers, our encrypted networks include a Batch Normalization Layer. Since these layers are both linear transformations, they can be easily collapsed:

$$y_{fc+bn} = \gamma\frac{(A\cdot x+b)-\mu}{\sqrt{\sigma^2+\epsilon}} + \beta = A'\cdot x + b', \tag{5}$$

where A and b are the weights and bias of the FC layer, and γ, β, μ and σ are the scale, bias, mean and standard deviation of the BN layer, while ϵ is a small constant for numerical stability. In this way, the weights of the FC and BN only need to be scaled once. However, the activation layer, being a polynomial, includes multiplicative coefficients, which also need to be scaled. Therefore, to avoid increasing the degree of the scaling factor, we can instead expand the squared term of the polynomial, and pre-compute every constant, obtaining:

$$y_{fc+bn+act} = (A''\cdot x)^2 + B''\cdot x + c'' \tag{6}$$

where A'' and B'' are matrices with the same dimensions as A and c'' is a bias vector. The final discretized equation for a set of FC+BN+Act layers then becomes:

$$y_{fc+bn+act} = (\lfloor s * A''\rfloor \cdot x)^2 + \lfloor s^2 * B''\rfloor \cdot x + \lfloor s^2 * c''\rfloor, \tag{7}$$

[1] This value depends on the plaintext modulus.

Method	Cold			Depression (Class.)			Depression (Regr.)		Parkinson's Disease		
	F1 Score (%)	Precision (%)	Recall (%)	F1 Score (%)	Precision (%)	Recall (%)	RMSE	MAE	RMSE	MAE	ρ
Baseline ENN	56.1	63.2	54.9	55.4	59.9	59.6	6.69	5.59	16.1	12.7	0.43
QNN	53.0	56.7	66.8	60.3	60.2	60.6	6.74	5.62	15.9	12.6	0.53
Scaled QNN	50.2	56.5	66.9	59.8	60.0	60.5	6.67	5.63	15.8	12.6	0.52

Table 1. Results for Paralinguistic Tasks.

where s is the scaling factor.

From these equations, we get that the degree of the scaling factor at the exit of the first FC+BN+Act layer set will be 2, for the second layer it will be 6, for the third layer 14, and so on. For this reason, considering the maximum value allowed, referenced above, if we have a scaling factor of 100, the depth of the network can be at most 2, otherwise during the encrypted inference values will not yield a correct decryption.

4.6. Network Architecture and Training

The network used in our experiments follows a generic architecture, consisting on two FC layers, each followed by a BN and an Activation layer, together with a final output FC layer. The first and second FC layers have 150 and 50 hidden nodes, respectively, while the output FC layer has only one, for both classification and regression tasks. For classification tasks, an output Activation layer was included after the third FC layer. During training, a dropout layer was also inserted, before the second and third FC layers [24]. While the first and second activation layers were polynomial approximations, the output activation layer introduced for classification tasks is not. Instead, a real Sigmoid was used. Due to the low complexity of the computation of a Sigmoid, we found that using this activation layer was a positive trade-off between the accuracy of the model and its privacy.

The size of our network was limited to three sets of layers for three reasons: on one hand, the datasets used for our experiments are relatively small, and using larger networks did not yield any performance gains; on the other hand, adding another layer set (FC + BN + Activation) would require an increase in the encryption parameters, resulting in a higher computational toll; finally, including a third set of layers would cause values in the Scaled QNN to grow larger than 2^{60}, which would result in incorrect predictions.

All networks were implemented and trained using Keras [25], with Tensorflow as backend. The models were trained with a learning rate of 0.02, 1,000 epochs with early stopping, and a weight decay of 0.005, using RMSProp. As loss functions, we used Binary Cross-Entropy (BCE) for classification, and Mean Square Error (MSE) for regression.

5. RESULTS

Previous experiments with other speech tasks showed us that using 4-bit feature quantization and a scaling factor of $s = 150$ yielded the best trade-off between accuracy, and the maximum absolute value of the network's computations. In addition, with this factor, the largest value in the network never surpasses 2^{55}.

Using these values we were able to obtain the results presented in Table 1, for the three datasets described above. We can see that there is a slight improvement from the baseline ENN to the QNN for all four tasks. However, from the QNN to the Scaled QNN, we can observe some degradation. While this was to be expected, since by scaling the network's weights we are limiting their precision, it was interesting to see how in some cases the use of quantization helped the network achieve better results. We hypothesize that quantizing the input features removes some of their noise and variability, and thus functions as a regularization technique, which may help the model to generalize better, and to achieve better results on unseen data.

When considering computational cost, the implementation of the ENN and the QNN takes 4.5s to compute a single encrypted prediction, while the Scaled QNN takes 23s. However, since 16,384 predictions can be made at the same time, the effective value for a prediction in the Scaled QNN is 1.4 milliseconds.

6. CONCLUSIONS

In this work we focused on discretizing an NN and its inputs using feature quantization and weight scaling, in order to apply an LHE *batching* technique. Our results showed that this can be done with minimal accuracy degradation, and we were able to perform 16,384 simultaneous predictions. However, this comes at the cost of having a maximum number of 2 activation layers in the network, and an increase in computational cost, as a single batch takes around 23 seconds to compute. Nevertheless, the effective time cost of a single prediction is 1.4 milliseconds as opposed to the original approach where a prediction took on average 4.5 seconds to compute.

As future work, it would be interesting to explore other quantization functions to be applied to the input features, as well as other weight discretization techniques that would not require the use of a scaling factor. Furthermore, our approach relies on the client to compute the input features of the network. Since we are considering the setting of a Machine Learning as a Service Application, it would be important to minimize the role of the client in the computation process. A potential way to do this would be research secure frameworks that could used to develop *end-to-end* networks.

7. REFERENCES

[1] M. A. Pathak and B. Raj, "Privacy-Preserving Speaker Verification and Identification Using Gaussian Mixture Models," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 2, Feb 2013.

[2] J. Portêlo, A. Abad, B. Raj, and I. Trancoso, "Secure Binary Embeddings of Front-end Factor Analysis for Privacy Preserving Speaker Verification," in *INTERSPEECH*, 2013, pp. 2494–2498.

[3] A. Jiménez, B. Raj, J. Portêlo, and I. Trancoso, "Secure Modular Hashing," in *Information Forensics and Security (WIFS), IEEE International Workshop*, 2015.

[4] M. Dias, A. Abad, and I. Trancoso, "Exploring Hashing and Cryptonet based Approaches for Privacy-preserving Speech Emotion Recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2018 IEEE International Conference*, 2018.

[5] F. Teixeira, A. Abad, and I. Trancoso, "Patient privacy in paralinguistic tasks," in *Proceedings Interspeech*, 2018, pp. 3428–3432.

[6] R. Gilad-Bachrach, N. Dowlin, and K. Laine et al., "CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy.," in *ICML*, 2016, vol. 48 of *JMLR Workshop and Conference Proceedings*, pp. 201–210.

[7] H. Chabanne, A. de Wargny, J. Milgram, and C. Morel et al., "Privacy-Preserving Classification on Deep Neural Network.," *IACR Cryptology ePrint Archive*, vol. 2017, pp. 35, 2017.

[8] E. Hesamifard, H. Takabi, and M. Ghasemi, "CryptoDL: Deep Neural Networks over Encrypted Data.," *CoRR*, vol. abs/1711.05189, 2017.

[9] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, "Deepsecure: Scalable provably-secure deep learning," in *55th ACM/ESDA/IEEE DAC*. IEEE, 2018.

[10] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *2017 IEEE Symposium on Security and Privacy (SP)*, 2017.

[11] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via MiniONN transformations," in *Proceedings of the ACM SIGSAC Conference*, 2017.

[12] M. S. Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar, "Chameleon: A hybrid secure computation framework for machine learning applications," in *Proceedings of the Asia Conference on Computer and Communications Security*. ACM, 2018.

[13] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *53rd Annual Allerton Conference*, Sept 2015.

[14] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) Fully Homomorphic Encryption without Bootstrapping," *ACM Trans. Comput. Theory*, July 2014.

[15] J. Krajewski, S. Schnieder, and A. Batliner, "Description of the Upper Respiratory Tract Infection Corpus (URTIC).," in *INTERSPEECH*, 2017.

[16] B. Schuller, S. Steidl, A. Batliner, and E. Bergelson et al., "The INTERSPEECH 2017 Computational Paralinguistics Challenge: Addressee, Cold & Snoring," in *Computational Paralinguistics Challenge (ComParE), Interspeech 2017*, 2017.

[17] J. Gratch, R. Artstein, G. M. Lucas, and G. Stratou et al., "The Distress Analysis Interview Corpus of human and computer interviews.," in *LREC*, 2014, pp. 3123–3128.

[18] M. F. Valstar, J. Gratch, B. W. Schuller, and F. R. et al., "AVEC 2016 - Depression, Mood, and Emotion Recognition Workshop and Challenge," *CoRR*, vol. abs/1605.01600, 2016.

[19] J. R. Orozco-Arroyave, J. D. Arias-Londoño, J. F. V. Bonilla, and M. C. Gonzalez-Rátiva et al., "New Spanish Speech Corpus Database for the Analysis of People Suffering from Parkinson's Disease.," in *LREC*, 2014.

[20] B. W. Schuller, S. Steidl, A. Batliner, and S. Hantke et al., "The INTERSPEECH 2015 Computational Paralinguistics Challenge: Nativeness, Parkinson's & Eating Condition.," in *INTERSPEECH*, 2015, pp. 478–482.

[21] A. Pompili, A. Abad, P. Romano, and I. P. Martins et al., "Automatic Detection of Parkinson's Disease: An Experimental Analysis of Common Speech Production Tasks Used for Diagnosis.," in *TSD*, 2017, Lecture Notes in Computer Science.

[22] F. Eyben, F. Weninger, F. Gross, and B. Schuller, "Recent developments in openSMILE, the Munich Open-source Multimedia Feature Extractor," in *Proceedings of the 21st ACM International Conference on Multimedia*, 2013.

[23] K. Laine, H. Chen, and R. Player, "Simple Encrypted Arithmetic Library - SEAL v2.3.1," Tech. Rep., Microsoft, 2017.

[24] N. Srivastava, G. E. Hinton, A. Krizhevsky, and I. Sutskever et al., "Dropout: A Simple Way to Prevent Neural Networks from Overfitting.," *Journal of Machine Learning Research*, 2014.

[25] F. Chollet et al., "Keras," `https://github.com/keras-team/keras`, 2015.