

# Large-Scale Recurrent Neural Networks with Fully Homomorphic Encryption for Privacy-Enhanced Speaker Identification

Anonymous

Redacted

Redacted

Redacted

Redacted

Anonymous

Redacted

Redacted

Redacted

Redacted

**Abstract**—Temporal classification tasks such as speaker identification are often performed by recurrent neural networks (RNNs) that observe potentially private (sensitive) data in order to provide service. Although encrypting this data safeguards it during storage and transit, decryption for computation introduces a potential vulnerability. Fully homomorphic encryption (FHE) is a privacy enhancing technology that supports computation over encrypted data. A neural network with multiple RNN layers and attention over encrypted data for this task is presented. Using GPU acceleration and novel contributions: (1) a RNN quantization procedure with ternarized parameters and binarized activations, and (2) a ciphertext-ciphertext multiplication method for attention that reduces required computation by 50%, yields the first published multi-layer RNN with attention over encrypted data. This marks a significant step toward practical privacy-enhanced temporal classification.

**Index Terms**—speaker identification, fully homomorphic encryption, recurrent neural networks, attention, quantization

## I. INTRODUCTION

Over the past two decades, machine learning as a service has surged into a global industry. With advancements in computer hardware and the proliferation of rich, accessible data, there is a rising demand for advanced tools like AI-driven chatbots and personal assistants which amplifies the importance of ensuring data privacy during storage, transmission, and computation. Fully homomorphic encryption (FHE) is a privacy enhancing technology that can be used to protect data during computation, allowing the same services mentioned above to be performed over encrypted data. For instance, FHE enables the ability to send encrypted samples of speech to third-party AI-driven voice assistants, with the assurance that the data can never be observed in the clear, and receive an encrypted answer (e.g. speaker ID) that only the holder of the private key can decrypt.

A significant body of literature focuses on the execution of neural networks over encrypted data, such as convolutional neural networks and simple logistic regression [1]. However, there is limited research in the area of recurrent neural networks (RNNs) [2, 3]. The characteristics of RNNs that make them powerful over sequential data and good models for temporal classification tasks (TCTs) serve as limitations in most FHE schemes. Primarily, RNNs contain deep recurrence relationships. In FHE, security relies on the presence of noise in each ciphertext [4], which grows with each consecutive mathematical operation, and can become large enough to prevent proper decryption. Consequently, even short RNN architectures are difficult to implement. Secondly, FHE operations are orders of magnitude slower than their plaintext counterparts, making RNNs, which are usually latency-sensitive, simply too inefficient to be useful.

Considering the widespread use of RNNs and *attention* to model end-to-end TCTs [5–7], this work centers on providing a path towards efficient large-scale attention-based RNN evaluation over encrypted data *non-interactively*, where the server performs the full inference without any aid from the client. We focus on the CGGI [8] mathematical foundation for FHE since it possesses an efficient bootstrapping method (an operation that reduces accumulated noise in ciphertexts). This enables unlocking limitless depth, a crucial feature for RNNs, while minimizing latency. The bootstrapping operation can also be accelerated by GPUs [9]. Using CGGI necessitates exploring quantization methods for RNNs, a challenging task due to the inherent amplification of noise along the time dimension in RNNs. Furthermore, since CGGI does not contain native ciphertext-ciphertext multiplication, and the attention model [10], which serves as an important aggregation layer in RNNs, relies on multiplication between activations to measure relative importance, there remains an additional challenge of finding a practical way to multiply ciphertexts.

To address these challenges, we present the **four-step quantization procedure for RNNs**. The procedure utilizes quantization-aware training to sequentially transform a full-precision vanilla RNN into a quantized version with binarized activations ( $\pm 1$ ) and ternarized parameters ( $0, \pm 1$ ). The procedure employs a novel method for calculating ternarization thresholds and utilizes an innovative gradient scaling technique to address vanishing/exploding gradient issues, preventing degradation of successful RNN quantization. We also present a **novel multiplication method between ciphertexts** that encrypt values in  $\{-1, 1\}$ . While the state of the art [11] uses two bootstrapping operations, our method uses only one due to the simplified domain of encrypted values.

We evaluate the speaker identification task over the FHE-encrypted VoxCeleb1 [12] test dataset using a 12.6M parameter, multi-layer, stacked vanilla RNN with multi-headed self-attention. Our model achieves superior latency while supporting two orders of magnitude more parameters than the state of the art for RNN evaluation [2]. It also processes encrypted inputs between 130 and 426 timesteps (1-4 s of speech), an order of magnitude larger time window than [2].

This paper is organized as follows. Section II provides background information. Section III introduces two major contributions. Section IV discusses experimental results, and section V compares our methods to the state of the art.

## II. BACKGROUND

### A. Fully Homomorphic Encryption over the Torus

Fully homomorphic encryption (FHE), pioneered by Gentry [13], is a class of encryption technologies that allow unlimited compu-

tation on encrypted data. Prior to Gentry’s breakthrough, various schemes limited operations to either addition or multiplication, or constrained the number of consecutive operations on ciphertexts [14]. Gentry’s scheme allowed both addition and multiplication, and his development of a *bootstrapping* procedure to decrease the noise in a ciphertext unlocked the possibility of an unlimited number of consecutive arithmetic operations. FHE security relies on the hardness of the Learning With Errors (LWE) problem [4], foundational in lattice cryptography, with its strength set by the security parameter  $\lambda$ . Encrypted messages contain sampled noise that is large enough to guarantee LWE hardness and small enough to ensure correctness after decryption. However, every operation increases noise, potentially making correct decryption impossible without bootstrapping.

Bootstrapping is the most computationally expensive FHE operation. The CGGI [8] foundation of FHE provides one of the fastest bootstrapping procedures, known as programmable bootstrapping (PBS) [11]. PBS reduces noise in ciphertexts while evaluating a lookup table (LUT) representing any arbitrary function over signed integers. This is critical in neural networks, enabling non-linear activation functions and allowing unlimited depth in RNNs due to noise reduction. To correctly construct a LUT, the *negacyclic* property is key. For example, consider the sign function,

$$\text{sign}(x) = \begin{cases} +1 & \text{if } x \geq 0, \\ -1 & \text{otherwise.} \end{cases} \quad (1)$$

For a function  $f(x)$  over  $p$ -bit signed integers where  $x \in [0, N-1]$  and  $N = 2^{p-1}$ , PBS evaluates  $f(x)$  for  $x \in [0, N-1]$  and  $-f(x)$  for  $x + N$ . The LUT for the sign function is constructed with  $N$  values,  $[f(0) = 1, \dots, f(N-1) = 1]$ , and PBS extends it to  $2N$  values, where  $f(x + N) = -f(x)$ , covering  $x \in [-N, N-1]$ .

CGGI supports ciphertext addition, ciphertext-plaintext scalar multiplication, and LUT evaluation. In our setup, clients FHE-encrypt their input data and send it to the server, which keeps inputs and activations encrypted while using plaintext model parameters and FHE operations to perform inference. The server returns the encrypted results for client decryption, ensuring only the input data remains private. CGGI supports (un)signed integers up to eight bits to maintain precision and efficiency [15]. Using larger bit-widths causes inaccuracies unless CGGI parameters are adjusted, but increasing the parameters to correct this reduces efficiency. Combined with its lack of right-shift and division support, this **requires low-bit, non-fixed-point integer quantization** for RNN evaluation over encrypted data.

### B. Quantization of RNNs for CGGI Evaluation

In neural networks, quantization is generally employed to reduce memory storage and improve computational efficiency [16]. Our use case is one of necessity. Several studies that evaluate neural networks using CGGI, exclusive of RNNs, employ various quantization methods. For instance, REDsec [17], TAPAS [18], and FHE-DiNN [19] utilize binarization for activation quantization, transforming values into  $\pm 1$  using the sign function defined in Equation 1. TAPAS and REDsec use ternarization for parameter quantization, mapping values to  $\{-1, 0, 1\}$ . Equation 2 defines a method to ternarize  $x \in \mathbb{R}$  given a ternarization threshold  $\tau \in \mathbb{R}_{>0}$ .

$$\text{ternary}(x, \tau) = \begin{cases} 0, & \text{if } |x| \leq \tau, \\ \text{sign}(x), & \text{otherwise.} \end{cases} \quad (2)$$

Simply transforming all floating-point (FP) inputs, activations, and parameters in a neural network to binary or ternary values after training introduces substantial quantization noise (i.e.,  $|x - \text{sign}(x)| \gg 0$

for  $|x| \gg 1$  or  $|x| \ll 1$ ), which can lead to significant accuracy loss [16]. RNNs, in particular, contain a feedback loop that can magnify any added noise with each additional timestep, further intensifying the problem of quantization noise. This issue is theoretically demonstrated by LAB [20], which proves that binarization without scaling causes RNNs to explode along the time dimension. Quantization-aware training (QAT), as utilized by REDsec and TAPAS, addresses this issue by injecting quantization noise during the forward step in training. This process introduces non-differentiable nodes that quantize and immediately dequantize values. Straight-through estimation, a part of QAT, allows gradients to pass through these nodes during backpropagation, enabling the network to maintain its parameters, activations, and gradients in FP. This approach allows the network to learn effectively and generalize to the presence of any quantization noise, thereby retaining accuracy despite the added noise [16].

While QAT can mitigate quantization noise, vanishing and exploding gradients in RNNs remain challenging. Consequently, RNN quantization research focuses on fixed-point quantization [21] and incorporating normalization layers after quantization [22]. However, normalization layers convert quantized integers back to FP, which CGGI does not support. Fixed-point arithmetic lacks support unless each  $p$ -bit value is represented by a ciphertext per bit [2, 17, 18]. This allows simple right-shift operations but results in poor performance due to complex circuits with numerous PBS operations over ciphertext arrays. Thus, we seek a quantization method for RNNs that: (1) enables single-ciphertext representation of inputs and activations during evaluation, allowing one PBS per activation and enhancing efficiency; (2) employs QAT to retain accuracy despite significant quantization noise; and (3) leverages low-bit quantization to minimize the required bit-width of the message space during accumulation. To our knowledge, no such method exists. Section III-A presents an algorithm that meets these objectives.

## III. CONTRIBUTIONS

### A. A Novel Quantization Procedure for RNNs

The **four-step quantization procedure** sequentially quantizes a (un)trained vanilla RNN by binarizing activations, ternarizing inputs, and ternarizing parameters through QAT at each step. Algorithm 1 details the procedure. We discuss two important features below.

**1) Ternarization threshold:** As defined in Equation 2, ternarization involves setting a threshold  $\tau$ . Algorithm 1 implements ternarization at the *layer* level, where  $\tau_l$  is computed over a set of values  $\mathcal{X}$  in each layer  $l$ , which, for instance, can represent the input dataset  $\mathcal{D}$  for the input layer, or the set of model parameters  $\theta_l$  in layer  $l$ . Liu *et al.* [24] suggest setting  $\tau_l = 0.75 \cdot \mathbb{E}(|\mathcal{X}|)$ , where  $\mathbb{E}(\cdot)$  is the expected value, and  $|\cdot|$  is the absolute value. Conversely, Zhu *et al.* [25] propose setting  $\tau_l = t \cdot \max(|\mathcal{X}|)$ , with  $t$  as a constant scalar hyperparameter. Our novel approach combines these methods by setting:

$$\tau_l = t \cdot \mathbb{E}(|\mathcal{X}|), \quad (3)$$

where  $t$  is the **ternarization scale**, a hyperparameter in Algorithm 1. This generalization provides flexibility in adjusting  $\tau_l$ , as we have empirically found that varying  $t$  with the expected value, rather than the maximum, improves model performance.

**2) Gradient scaling:** Inspired by straight through estimation [16], we adopt a custom activation function that employs the sign function (Equation 1) during the forward pass and the tanh derivative during backpropagation, termed *sign-with-tanh-derivative* in Algorithm 1. This approach facilitates effective quantization of values while maintaining a smooth loss landscape during training. However, the

**Algorithm 1:** Four-Step Quantization for RNNs

**Input :** Vanilla RNN model  $M_0$ , trained or untrained, input dataset  $\mathcal{D}$ , ternarization scale  $t$ .

**Output:** Quantized RNN with binary activations and ternary inputs/parameters  $M_4$ .

**Step 1: Train a Vanilla RNN**

- 1 Change activations in  $M_0$  to tanh.
- 2  $M_1 \leftarrow \text{train}(M_0)$ .

**Step 2: Binarize Activations**

- 3 Change activations in  $M_1$  to *sign-with-tanh-derivative*.
- 4  $M_2 \leftarrow \text{train}(M_1)$ .
- 5 **if** *exploding/vanishing gradients* **then**
- 6     Increase batch size.
- 7     Repeat lines 3 - 4.
- 8 **if** *model is still not training* **then**
- 9     **foreach** RNN layer in  $M_3$  **do**
- 10         Set temperature scale  $s_l$  for gradient scaling.
- 11         Repeat line 3.
- 12         Repeat line 4 while applying gradient scaling.

**Step 3: Ternarize Inputs**

- 13  $\mathcal{D} \leftarrow \text{ternary}(\mathcal{D}, \tau_I)$  where  $\tau_I = t \cdot \mathbb{E}(|\mathcal{D}|)$ .
- 14  $M_3 \leftarrow \text{train}(M_2)$ .

**Step 4: Ternarize Model Parameters**

- 15 **foreach** layer  $l$  in  $M_3$  w/ parameter distribution  $\theta_l$  **do**
- 16      $\tau_l \leftarrow t \cdot \mathbb{E}(|\theta_l|)$      /\* eq. 3 \*/
- 17 **foreach** RNN layer  $l$  in  $M_3$  **do**
- 18     Set temperature scale  $s_l$  for gradient scaling.
- 19  $M_4 \leftarrow \text{train}(M_3)$  while applying (1) gradient scaling and (2) ternarization in the forward step.
- 20 **if** *model fails to train* **then**
- 21     Increase number of parameters in each layer of  $M_0$  by  $3x$ , as suggested by *Mishra et al.* [23].
- 22     Repeat lines 1 - 19.

sign function can introduce significant quantization noise, as noted in section II-B. To mitigate exploding or vanishing gradients within the constraints of CGGI evaluation, we introduce a novel **gradient scaling** technique. Unlike conventional methods that scale both forward and backward computations [26], our innovation lies in exclusively applying a scaling factor  $1/s$  to the affected gradients during backpropagation, while preserving the integrity of forward step values. Here, the scaling factor  $1/s$ , where  $s$  is referred to as the **temperature scale**—a hyperparameter in Algorithm 1—is applied to the gradients of all the pre-activations in the RNN layer. This method effectively stabilizes gradients without compromising the network’s adherence to binary activations, essential for CGGI evaluation.

Algorithm 1 results in a quantized RNN with binary activations and ternary inputs and parameters, effectively limiting the required bit-width for precise pre-activation representation. This approach enables a single ciphertext representation and the use of a single PBS operation per activation during encrypted inference. Moreover, integrating QAT ensures robust model performance, thereby successfully fulfilling the objectives outlined in section II-B. In the following section, we investigate a way to multiply ciphertexts to enable encrypted attention, an important aggregation layer in RNNs.

**B. Multiplication of Binarized Ciphertexts**

Within CGGI, the external product operation facilitates the multiplication of LWE ciphertexts with those of a different encryption type, specifically GGSW [8]. To perform multiplication between LWE

TABLE I: **Binary Multiplication Truth Table.** Refer to equation 4 for a definition of  $m$ .

$x$	$y$	$x - y$	$m \cdot (x - y)$	$x \cdot y$
+1	+1	0	0	+1
+1	-1	+2	+2m	-1
-1	+1	-2	-2m	-1
-1	-1	0	0	+1

ciphertexts, circuit bootstrapping (CBS) must be used to convert an LWE ciphertext to a GGSW ciphertext. However, CBS is a computationally expensive operation [8], and would dramatically increase the execution time of circuits that are abundant in multiplications. In [11], an inner product between LWE ciphertexts is proposed, called BFV-like multiplication, though noisy. Using a mathematical identity, the authors also propose a general way of multiplying two LWE ciphertexts through two PBS operations. While this method can be performed across a much larger message domain, limiting the domain to  $\{-1, 1\}$  offers some advantages.

We present a **novel multiplication method between binarized ciphertexts that requires only one PBS**. As shown in the third column of Table I, subtracting two binary values  $x, y \in \{-1, +1\}$  results in three distinct values  $\{0, +2, -2\}$ . As a result, the third column can be mapped to the last column by a LUT ( $x - y \rightarrow x \cdot y$ ) and used to evaluate a single PBS. However, the values in column 3 are numerically close with respect to the  $p$ -bit plaintext message space. Since ciphertexts contain noise, accumulation of noise from operations prior to multiplication could build and become large enough to offset  $x - y$  by a small amount, causing an incorrect output. Scaling by  $m$  from Equation 4 separates the values, as shown in column 4. This allows  $2m$  or  $-2m$ , for example, to align with the centers of the second and third regions in  $\text{LUT}_{\text{mult}}$ , as defined in Equation 5, respectively. These regions contain repetitions of the correct output value, ensuring error robustness.

$$m = 3 \cdot 2^{p-5} \quad (4)$$

$$\text{LUT}_{\text{mult}} = [\{+1\}_{N/4}, \{-1\}_{N/4}, \{+1\}_{N/4}, \{-1\}_{N/4}] \quad (5)$$

where  $p \geq 5$  is the bit-width of the plaintext message space,  $N = 2^{p-1}$ , and  $\{x\}_r = x$  repeated  $r$  times. In our implementation, we utilize an 11-bit message space, resulting in  $m = 192$ .

**IV. EXPERIMENTAL RESULTS**

We evaluate the SpeakerID RNN (Figure 1) for the speaker identification (SID) task over the VoxCeleb1 [12] test set. VoxCeleb1 is considered a large-scale dataset for various speaker tasks, including SID (a multi-class problem to classify speakers from raw audio). VoxCeleb1 contains 1251 speakers and 153,516 utterances recorded in-the-wild. On average, there are 123 utterances per speaker, each 8.2s long. Drawing inspiration from [7, 12], each speech sample is preprocessed into log-mel spectrograms using 80 mel-scale bins, 25 ms frames, a 10 ms stride, and a 16 kHz sampling rate. It is then normalized, stacked by four frames, and downsampled by a factor of 3 to create input vectors with 320 elements for each timestep. We utilize the default VoxCeleb1 data split [12]. During each training epoch, 4s samples are drawn at random from each full length sample. Validation is performed similarly, while testing is performed over full length samples. We measure top-1 and top-5 accuracy to evaluate model performance as in [12].

Motivated by [7], the **SpeakerID RNN model** (Figure 1) contains two RNN layers, with the outputs of the first RNN downsampled by a factor of 2. Inspired by [27], a multi-headed, self-attention module, comprised of two feed-forward (FF) layers, is applied to all of the second RNN outputs to perform aggregation. The last FF layers

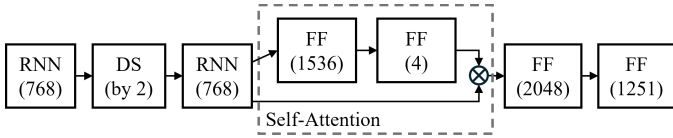


Fig. 1: *The SpeakerID RNN model.* DS (by 2) refers to downsample by a factor of 2, FF refers to feed-forward. Numbers underneath FF and RNN refer to number of units. Consists of 12,644,352 parameters.

classify speakers based on the flattened outputs from the attention module. The model activations are tanh, the outputs are fed into a softmax, and cross-entropy loss is used as the training objective. Adam optimization and a batch size of 512 are used. In step 1 of Algorithm 1, default parameter initialization settings for RNNs are used and we apply the learning rate scheduler from [28] with 5400 warmup steps, a model size of 1536, and  $\mathcal{L}_2$  regularization at  $10^{-4}$ . Steps 2-3 use a fixed learning rate of  $10^{-4}$ , while step 4 switches to a cosine schedule starting at  $2 \cdot 10^{-6}$ , decaying by  $10^{-1}$  after 100 epochs, with no regularization. TensorFlow v2.10.1 and QKeras v0.9 [29] are used to train, quantize, and evaluate plaintext models. For encrypted evaluation with CGGI, we use a customized Concrete-Core library which includes bug fixes enabling multi-GPU support [9]. A machine with two AMD EPYC 7763 CPUs, two Nvidia A100 GPUs, 512 GB RAM, and Ubuntu v22.10 performs our evaluations. CGGI parameters from [15] ensuring 128-bit security are used, with an 11-bit plaintext message space to handle large accumulations. To prevent overflow in the final layer, we divide each dot product into 64 smaller summations. After decryption, the plaintexts are summed and fed into a softmax. The source code is available [30].

Table II presents the plaintext test accuracy results (columns 2-3) for the SpeakerID RNN after each step in Algorithm 1. We utilize a ternarization scale  $t = 1.6$  for parameters,  $t = 0.7$  for inputs, and a temperature scale  $s = 5$  when required. The results in step 1 showcase the full precision accuracy, which achieves comparable (1.54% better) top-1 accuracy than [12] over the same dataset and SID task. Performing the next three steps in the quantization procedure results in decreases in top-1 and top-5 accuracies of -25.04% and -14.47%, respectively—a promising significance result since this is the **first successful quantization of RNNs using binary activations and ternary parameters without scaling of weights/activations and normalization, allowing the arithmetic to be purely integer and not fixed-point**. Furthermore, the positive change between start and end validation accuracies in each step (column 4) emphasizes the importance of step-wise quantization with QAT. In contrast, combining steps 2-4 into a single step yields only 12.94% accuracy, approximately 30% less than using the four-step approach. Thus, using all four steps in sequence significantly enhances accuracy compared to commonly used single-step quantization.

Using the quantized parameters from step 4, the evaluation over encrypted data yields encrypted and plaintext accuracies of 28% and 53% for top-1, and 44% and 78% for top-5, respectively, over the encrypted VoxCeleb1 test set. The decreased encrypted accuracy can largely be attributed to integer accumulation overflowing the plaintext modulus in the FF(2048) layer. Notably, for a correctly classified, 1.9s (188 timesteps) utterance, our evaluation exhibits a latency of 531s. In comparison, the one-layer, 300-unit, 25-timestep RNN (without attention) in SHE [2] contains approximately 180K parameters and exhibits a latency of 576s. Our model, with two orders of magnitude more parameters than SHE, successfully evaluates encrypted utterances between 130 and 426 timesteps (an order of magnitude more timesteps than SHE) and yet achieves

TABLE II: *Accuracy results of SpeakerID RNN over VoxCeleb1 [12] with the four-step quantization process.*  $\Delta_{\text{Top-1}}$  refers to the gain in top-1 validation accuracy from the beginning of the step to the end for an example training run.

Step	Test Acc.		Val. Acc.
	Top-1	Top-5	$\Delta_{\text{Top-1}}$
1	82.04%	92.52%	-
2	77.36%	90.28%	+31.67%
3	74.16%	88.64%	+7.77%
4	57.00%	78.05%	+17.20%

superior latency. This achievement is largely attributed to our use of a single-ciphertext representation, significantly reducing computation requirements. Moreover, we observe a linear drop in latency as the number of GPUs increases. Concrete-Core distributes PBS operations across multiple GPUs, with each GPU performing an equal amount of PBS operations, in parallel, to its number of streaming multiprocessors (SMs). The A100 GPU contains 108 SMs, and empirically, we found that a single A100 can perform 100 PBS operations in 75 ms. Thus, about 2888 GPUs would be required to parallelize all groups of PBS operations in the 1.9s sample above, bringing the latency down from 531s to an estimated 25s, an order of magnitude from plaintext latency.

## V. RELATED WORK

Only two studies have explored non-interactive evaluation of RNNs over encrypted data [2, 3], while none incorporate attention mechanisms. SHE [2] investigates a single-layer, 300-unit RNN with 180,000 parameters over 25 timesteps using the Penn Treebank dataset [31], relying on a fixed-point quantization scheme. This requires multiple ciphertexts per value, leading to high latency due to complex circuits, despite achieving good accuracy. In contrast, *Podschwadt et al.* [3] employ the CKKS FHE scheme [32], which eliminates the need for quantization, but the high cost of bootstrapping forces partitioning RNNs along the time dimension, disrupting feedback loops and resulting in 19.5 minutes of latency per sample, unsuitable for real-time tasks. Other studies have explored neural networks over encrypted data [17–19], though not specifically focusing on RNNs. TAPAS [18] and REDsec [17] use fixed-point quantization, requiring multi-ciphertext representations and resulting in significant latency. FHE-DiNN [19] uses post-training quantization, but is ill-suited for RNNs due to quantization noise amplified by feedback loops, which static methods fail to mitigate.

## VI. CONCLUSION AND FUTURE WORK

To move towards practical temporal classification over encrypted data, this paper introduces a four-step quantization method for RNNs. Successfully applied to the SpeakerID RNN, a large-scale, multi-layer vanilla RNN with attention, the method employs pure integer arithmetic (non-fixed-point), binary activations, and ternary inputs/parameters. For efficient attention evaluation over encrypted data, a novel ciphertext multiplication method between binary ciphertexts is proposed, yielding a 50% reduction in latency. Using these methods and the CGGI FHE scheme, we evaluate speaker identification on the encrypted VoxCeleb1 dataset, achieving superior latency over the state of the art [2]. Notably, the evaluation involves a model with two orders of magnitude more parameters and an order of magnitude more timesteps. This paper marks the first evaluation of: (1) speaker identification over encrypted data using RNNs, and (2) RNNs with attention over encrypted data. Future work will address message space overflow affecting encrypted accuracy.

## REFERENCES

- [1] R. Podschwadt, D. Takabi, P. Hu, M. H. Rafiei, and Z. Cai, "A survey of deep learning architectures for privacy-preserving machine learning with fully homomorphic encryption," *IEEE Access*, vol. 10, pp. 117 477–117 500, 2022.
- [2] Q. Lou and L. Jiang, "SHE: A Fast and Accurate Deep Neural Network for Encrypted Data," in *NeurIPS 2019*, ser. Advances in Neural Information Processing Systems, vol. 32, 2019. [Online]. Available: <https://proceedings.neurips.cc/paper/2019/file/56a3107cad6611c8337ee36d178ca129-Paper.pdf>
- [3] R. Podschwadt and D. Takabi, "Non-interactive Privacy Preserving Recurrent Neural Network Prediction with Homomorphic Encryption," in *2021 IEEE 14th International Conference on Cloud Computing (CLOUD)*, Sep. 2021, pp. 65–70. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9582263>
- [4] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *37th ACM STOC*. Baltimore, MD, USA: Association for Computing Machinery, May 2005, pp. 84–93. [Online]. Available: <https://doi.org/10.1145/1060590.1060603>
- [5] H. Sak, A. Senior, K. Rao, O. İrsoy, A. Graves, F. Beaufays, and J. Schalkwyk, "Learning acoustic frame labeling for speech recognition with recurrent neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4280–4284.
- [6] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 4960–4964.
- [7] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shang-guan, B. Li, G. Pundak, K. C. Sim, T. Bagby, S. y. Chang, K. Rao, and A. Gruenstein, "Streaming End-to-end Speech Recognition for Mobile Devices," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6381–6385.
- [8] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "TFHE: Fast Fully Homomorphic Encryption Over the Torus," *Journal of Cryptology*, vol. 33, no. 1, pp. 34–91, Jan. 2020. [Online]. Available: <https://doi.org/10.1007/s00145-019-09319-x>
- [9] I. Chillotti, M. Joye, D. Ligier, J.-B. Orfila, and S. Tap, "CONCRETE: Concrete Operates on Ciphertexts Rapidly by Extending TfhE," in *WAHC 2020–8th Workshop on Encrypted Computing & Applied Homomorphic Cryptography*, vol. 15, 2020, for bug-fix version, use concrete-core from <https://github.com/irskid5/concrete-core>.
- [10] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., San Diego, CA, USA, May 2015. [Online]. Available: <https://arxiv.org/abs/1409.0473>
- [11] I. Chillotti, D. Ligier, J.-B. Orfila, and S. Tap, "Improved Programmable Bootstrapping with Larger Precision and Efficient Arithmetic Circuits for TFHE," in *Advances in Cryptology – ASIACRYPT 2021*, M. Tibouchi and H. Wang, Eds. Singapore, Singapore: Springer International Publishing, 2021, pp. 670–699. [Online]. Available: [https://link.springer.com/chapter/10.1007/978-3-030-92078-4\\_23](https://link.springer.com/chapter/10.1007/978-3-030-92078-4_23)
- [12] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: A Large-Scale Speaker Identification Dataset," in *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association*, F. Lacerda, Ed. Stockholm, Sweden: ISCA, August 2017, pp. 2616–2620, for default dataset split, see [tensorflow.org/datasets/catalog/voxceleb](https://tensorflow.org/datasets/catalog/voxceleb). [Online]. Available: <https://doi.org/10.21437/Interspeech.2017-950>
- [13] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, ser. STOC '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 169–178. [Online]. Available: <https://doi.org/10.1145/1536414.1536440>
- [14] P. Martins, L. Sousa, and A. Mariano, "A Survey on Fully Homomorphic Encryption: An Engineering Perspective," *ACM Computing Surveys*, vol. 50, no. 6, p. Article 83, 2017. [Online]. Available: <https://doi.org/10.1145/3124441>
- [15] L. Bergerat, A. Boudi, Q. Bourgerie, I. Chillotti, D. Ligier, J.-B. Orfila, and S. Tap, "Parameter Optimization and Larger Precision for (T)FHE," *Journal of Cryptology*, vol. 36, no. 3, p. 28, Jun. 2023, for used parameter set, see table 8, row 12 in <https://eprint.iacr.org/archive/2022/704/20220620:172436>.
- [16] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A Survey of Quantization Methods for Efficient Neural Network Inference," in *Low-power computer vision : improve the efficiency of artificial intelligence*, 1st ed. Boca Raton, FL: Chapman and Hall/CRC, 2022, pp. 291–326.
- [17] L. Folkerts, C. Gouert, and N. G. Tsoutsos, "REDsec: Running Encrypted Discretized Neural Networks in Seconds," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, San Diego, California, March 2023.
- [18] A. Sanyal, M. Kusner, A. Gascon, and V. Kanade, "TAPAS: Tricks to Accelerate (encrypted) Prediction As a Service," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, D. Jennifer and K. Andreas, Eds., vol. 80. PMLR, 2018, pp. 4490–4499. [Online]. Available: <https://proceedings.mlr.press/v80/sanyal18a.html>
- [19] F. Bourse, M. Minelli, M. Minihold, and P. Paillier, "Fast Homomorphic Evaluation of Deep Discretized Neural Networks," in *Advances in Cryptology – CRYPTO 2018*. Springer International Publishing, 2018. [Online]. Available: [https://doi.org/10.1007/978-3-319-96878-0\\_17](https://doi.org/10.1007/978-3-319-96878-0_17)
- [20] L. Hou, Q. Yao, and J. T. Kwok, "Loss-aware Binarization of Deep Networks," in *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, April 2017. [Online]. Available: <https://openreview.net/forum?id=S1oWlN9ll>
- [21] L. Hou and J. T. Kwok, "Loss-aware Weight Quantization of Deep Networks," in *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada, May 2018. [Online]. Available: <https://openreview.net/forum?id=BkrSv0lA->
- [22] L. Hou, J. Zhu, J. T. Kwok, F. Gao, T. Qin, and T. Liu, "Normalization Helps Training of Quantized LSTM," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2021, pp. 7346–7356.
- [23] A. Mishra, E. Nurvitadhi, J. J. Cook, and D. Marr, "WRPN: Wide Reduced-Precision Networks," in *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada, May 2018. [Online]. Available: <https://openreview.net/forum?id=B1ZvaeAZ>
- [24] B. Liu, F. Li, X. Wang, B. Zhang, and J. Yan, "Ternary Weight Networks," in *2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5. [Online]. Available: <https://doi.org/10.1109/icassp49357.2023.10094626>
- [25] C. Zhu, S. Han, H. Mao, and W. J. Dally, "Trained Ternary Quantization," in *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, April 2017. [Online]. Available: [https://openreview.net/forum?id=S1\\_pAu9xl](https://openreview.net/forum?id=S1_pAu9xl)
- [26] P. Wang, X. Xie, L. Deng, G. Li, D. Wang, and Y. Xie, "HitNet: hybrid ternary recurrent neural network," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS'18. Curran Associates Inc., 2018, p. 602–612.
- [27] N. N. An, N. Q. Thanh, and Y. Liu, "Deep CNNs With Self-Attention for Speaker Identification," *IEEE Access*, vol. 7, pp. 85 327–85 337, 2019.
- [28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," *ArXiv*, no. 1706.03762, 2017. [Online]. Available: <https://arxiv.org/abs/1706.03762>
- [29] C. N. Coelho, A. Kuusela, S. Li, H. Zhuang, J. Ngadiuba, T. K. Aarrestad, V. Loncar, M. Pierini, A. A. Pol, and S. Summers, "Automatic heterogeneous quantization of deep neural networks for low-latency inference on the edge for particle detectors," *Nature Machine Intelligence*, vol. 3, no. 8, pp. 675–686, Aug. 2021.
- [30] Anonymous, "redacted," 2024, for training and quantization, see redacted. [Online]. Available: Redacted
- [31] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a Large Annotated Corpus of English: The Penn Treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993. [Online]. Available: <https://aclanthology.org/J93-2004>
- [32] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic Encryption for Arithmetic of Approximate Numbers," in *Advances in Cryptology – ASIACRYPT 2017*, T. Takagi and T. Peyrin, Eds. Springer International Publishing, 2017, pp. 409–437.