Privacy-Preserving Federated Inference for Genomic Analysis with Homomorphic Encryption

Anish Chakraborty and Nektarios Georgios Tsoutsos

University of Delaware
{anishch, tsoutsos}@udel.edu

Abstract. In recent years, federated learning has gained significant momentum as a collaborative machine learning approach, particularly in the field of medicine. While the decentralized nature of federated learning provides greater security guarantees compared to traditional machine learning methods, it is still susceptible to myriad attacks. Moreover, as federated learning becomes increasingly ubiquitous in medicine, its use for classification tasks is expected to increase; however, maintaining patient data confidentiality remains a significant challenge, especially for genetic data. While previous works focus on utilizing Single Nucleotide Polymorphisms or numerical data, recent advances in gene sequencing amplify the importance of inference from DNA sequences. In this work, we introduce a novel framework for secure federated inference on nucleotide-based genotype data and provide a gateway to private inference through fully homomorphic encryption. A federated model with five local clients was created and trained before being encrypted with the TFHE cryptosystem and placed for inference. We tested multiple different local and global model architectures, including a standard neural network, a support vector machine, and an LSTM (long-short-term memory), and compared their respective abilities across accuracy, precision, recall, and runtime metrics. These frameworks successfully identified promoter sequences encoded within given DNA sequences, as well as whether a given sequence was a coding sequence or an intergenomic sequence, showing their potential applications in secure genomic data analysis in a federated context. Our work represents a crucial step in privacy-preserving federated inference on nucleotide-based data.

Keywords: Federated Learning, Fully Homomorphic Encryption, Genomics, Privacy-Preserving Machine Learning.

1 Introduction

Federated learning provides a significant advantage for collaborative model training and inference, while also maintaining data security and privacy. While medical data is often abundant, it is often spread across multiple data centers. In these

cases, a decentralized, federated model can outperform conventional, centralized machine learning models. Furthermore, this architecture offers the ability to maintain the confidentiality of data, as local models are trained separately with their private data before the model's weights are aggregated to form a global model. This ensures that any confidential data is secure, even when being used to train a model with potential adversaries.

Although these models have numerous security benefits compared to a conventional centralized machine learning system, they remain vulnerable to security breaches. During inference, malicious adversaries may gain access to data as it is being transferred to the global model [23], severely hindering privacy guarantees with classified data. For instance, a man-in-the-middle attack (MITM), in which a malicious adversary secretly gains access to the data being sent for inference depicts a key issue with federated models. Alternatively, the global server itself could be corrupted, therefore, sidestepping the security which a decentralized architecture provides; consequently, the entire architecture would grow unusable, as adversaries could gain access to private, confidential data [3].

Recent advancements have demonstrated the feasibility of utilizing fully homomorphic encryption (FHE) as a solution to privacy breaches in federated learning. FHE schemes call for the user to encrypt the data before sending it to a server to perform computations; subsequently, the server can run its algorithm on this encrypted data, and then send the encrypted output back to the user, who can then decrypt it. Therefore, this scheme prevents adversarial attacks during data transfer, as well as attacks from a corrupt global server. Therefore, in the case of an MITM attack, the data and results would be encrypted under a secret key known only to the user, meaning an adversary would be unable to decrypt the data as it is being transferred for inference.

Many FHE-based federated inference works focus on imaging data and overwhelmingly utilize CKKS as their FHE scheme [32, 33]. However, while CKKS allows floating-point computations, it focuses on approximate computations [6], which reduces precision. While with images, reduced precision in image reconstruction can still lead to optimal results (allowing CKKS to be sufficient) [19], this lack of precision must be mitigated for meaningful results with nucleotide-based data. Our work surrounds the application of FHE-based federated learning to these nucleotide sequences to detect the presence of a specific promoter sequence; consequently, TFHE was used to allow for complete precision, for inaccuracies in a DNA sequence prove detrimental to determining its message. We note that our methodologies were tested with CKKS to further quantify the benefits and costs that stem from using either cryptosystem. Furthermore, other related literature utilizes numeric gene expression data [5, 26] or numeric encoding vectors of Single Nucleotide Polymorphisms (SNPs) [8, 12, 24, 28], but these data types do not contain the same requisite sequential precision that nucleotide data intrinsically does.

Despite the clear lack of pertinent research regarding FHE-based federated inference schemes in nucleotide data, this topic proves extremely important. While related works have demonstrated the feasibility of utilizing SNPs for en-

crypted inference, we are the first to present frameworks that can accurately perform encrypted inference on raw, DNA nucleotide data. As DNA sequencing becomes more prevalent, major data centers continue to gain access to DNA strings to use for predictive analytics. DNA sequences constitute a fundamental form of biological data; therefore, the ability to actively and accurately analyze them proves instrumental for rapidly identifying diseases or understanding biological processes such as protein formation. In this work, we introduce a novel framework (depicted in Figure 1) to perform federated classification tasks on DNA sequence data accurately. This work directly paves the way for future works in DNA classification, which can potentially be applied for disease detection, as well as a fuller understanding of the relationship between seemingly meaningless nucleotides and noticeable changes in expression. The ability to accurately detect promoter sequences will allow for a more clear understanding of the mechanisms behind the creation of proteins. Finally, using the TFHE encryption scheme, we ensure maximum precision with our data. The ability to accurately predict biologically significant outputs from encrypted inputs has significant clinical implications: our work clearly provides a framework for medical companies or agencies to cooperate, using genomic data to develop large models which can accurately perform inference on nucleotide data. This ability will allow medical firms to pervasively utilize DNA sequencing techniques to collaboratively perform inference, which has the potential to revolutionize biomedical research and disease detection. Overall, our contributions can be summarized by the following:

- We reveal a set of novel federated frameworks which can accurately perform inference on individual nucleotide string data, allowing hospitals and medical data centers to utilize genotype data in predictive analytics;
- We ensure complete data security during private inference without significantly compromising on precision by utilizing fully homomorphic encryption (FHE).
- We test our frameworks on multiple diverse public DNA datasets of varying feature lengths to demonstrate their novelty across varying bit widths and client counts.

Roadmap: The rest of the paper is organized as follows. Section 2 provides necessary preliminaries on homomorphic encryption, federated learning, and genetics. Section 3 discusses our proposed methodology. Section 4 presents our experimental results, as well as a brief discussion on the implications and limitations of our approach. Finally, Section 5 provides comparisons to related works, while Section 6 imparts our concluding statements.

2 Preliminaries

2.1 Fully Homomorphic Encryption (FHE)

Homomorphic Encryption (HE) is an encryption method that supports secure computation on encrypted data without first decrypting it. Specifically, modern

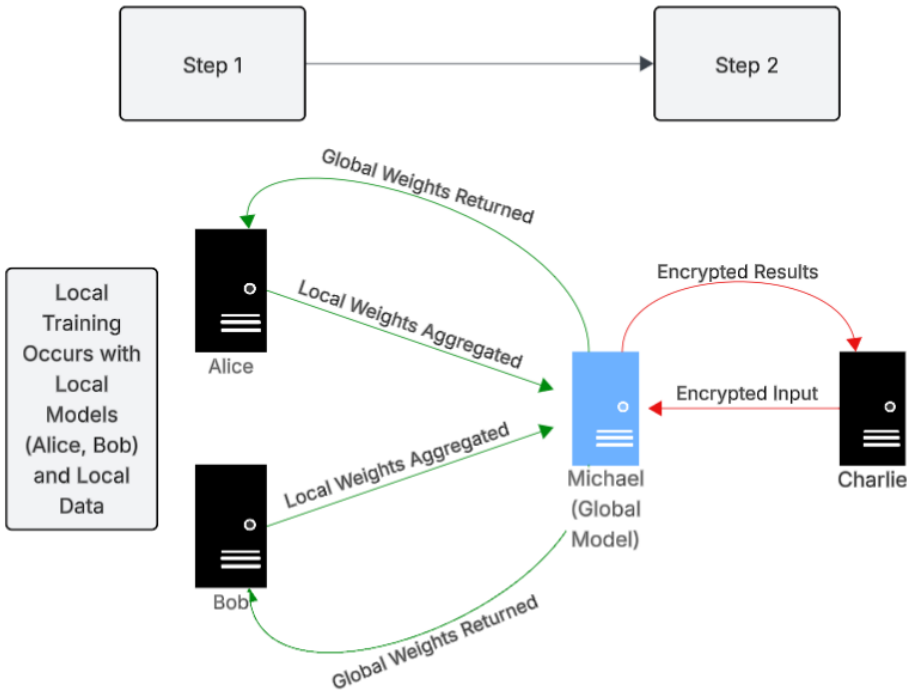

Fig. 1. Schematic of our proposed methodology. Step 1: Local models train on their respective local data, and perform aggregation with the global model. Step 2: Users can perform inference on this global model, and the outputs are protected through FHE.

HE systems can be split into three distinct subcategories: partial homomorphic encryption, leveled homomorphic encryption, and fully homomorphic encryption. Partial homomorphic encryption (PHE) refers to cryptosystems which allow for either multiplication or addition between two ciphertexts. Cryptosystems such as RSA [27], and Paillier [25] comprise notable examples of PHE. However, despite being able to perform unbounded additions or multiplications, PHE's inability to do both severely hinders its potential usefulness. Moreover, leveled HE (LHE) supports addition *and* multiplication on certain ciphertexts; however, this system is inefficient with larger algorithm depths. To ensure security, a small noise component is added to encrypted ciphertexts; consequently, repeated computations result in an aggregation of noise, which prevents the correct decryption of the ciphertext [16]. Intuitively, homomorphic addition results in linear increases of noise, while homomorphic multiplication results in exponential increases in noise. In 2009, Charles Gentry proposed the first implementation of a fully homomorphic encryption (FHE) scheme, which notably included a *bootstrapping* mechanism that significantly reduces noise levels [15]. Since the noise component

is the key facet to maintain security, bootstrapping helps ensure that noise is added without becoming overwhelming.

To elaborate, a bootstrapping mechanism calls for a decryption in the encrypted domain, which reduces the total amount of noise over a set of computations. However, the plaintext data is never exposed, as this computation happens completely in another level of encryption. Therefore, after applying the decryption algorithm homomorphically, the underlying message is not revealed, yet the amount of noise is reduced significantly. This creation of a bootstrapping mechanism intrinsically allows for FHE-based algorithms to run with unbounded depth; however, this bootstrapping method remains computationally intensive.

TFHE and CKKS: TFHE and CKKS remain two of the most commonly used cryptosystems in FHE, especially for neural network inference. The CKKS cryptosystem can perform floating-point computations, but this leads to approximate arithmetic [6]. On the contrary, TFHE is another popular FHE scheme that solely performs integer arithmetic and does not utilize approximations in its calculations [7]. As a result, the CKKS cryptosystem usually performs complex computations at a faster rate, compared to TFHE, though this speed may come at the cost of precision. Our work utilized both the TFHE and CKKS cryptosystems and noted the difference in model performance during encrypted inference.

2.2 Federated Learning

Federated learning constitutes a paradigm of computation and inference among multiple parties, and offers a decentralized approach to machine learning. Unlike conventional "centralized" machine learning, which has one sole node for training and updating weights, FL calls for multiple nodes of local networks, all of which train on separate data. Then, the gradients of these local models are aggregated onto one global model, which, consequently, has "knowledge" from the entirety of the local data [4]. This decentralized approach allows for medical companies to utilize potentially sensitive private patient data without disclosing it to rival companies or outsiders [10].

Despite federated learning offering a gateway to secure neural network computation with multiple parties, data must still be kept confidential when being sent to the models as well as during aggregation. Therefore, FHE has been deemed a valuable asset to ensure privacy in sensitive data during federated learning and inference [3], as this can prevent the prevalence of *data leakage* during model inference.

2.3 A Primer on DNA

DNA is the most fundamental building block in all living systems. Despite its ubiquity in all biological systems, its meaning proves elusive. Computational methods can be used to find patterns among DNA sequences. Given DNA's importance, documenting it is often spread across many different methods. Some methods utilize numeric data regarding the prevalence of a certain gene, while

other methods highlight specific single-nucleotide combinations (called single-nucleotide polymorphisms) which are known to have some significance, and use this data to generate regression models. However, DNA nucleotide data remains more unexplored. This type of data results from DNA sequencing, which determines the exact combination of nucleotides that make up a given strand of DNA [20]. Our work leverages this data for decentralized predictive analytics.

Single Nucleotide Polymorphisms: Single nucleotide polymorphisms (SNPs) are small fragments of DNA which capture known variations across individuals. These significantly shorter nucleotide combinations are known to embody some biological significance, and are often used in frameworks similar to our methodology. However, with the increase in DNA sequencing techniques, the importance of being able to rapidly identify patterns in raw, unfiltered DNA strings prove more important. As a result, this work performs computations on strings of DNA, embodying this new, novel use-case.

2.4 Promoters and Their Importance

The first dataset utilized in this work consists of detecting promoter sequences in strings of DNA nucleotide data. Promoters are a key part of DNA, as they help regulate transcription of RNA molecules. This concept is rooted in the central dogma of biology: for a protein to form, DNA instructions provide a blueprint for creating RNA sequences. This process is known as transcription. These RNA molecules can then undergo translation, which creates amino acids that later fold into proteins from an RNA blueprint. Therefore, this set of processes proves paramount for most biological organisms, describing the main transfer of information from DNA to build proteins. However, RNA polymerase, the molecule that creates RNA, needs to identify a promoter sequence within the DNA which is where it starts building the RNA. Finding these promoter sequences in a given strand of DNA paves the way towards understanding transcription. Furthermore, the processes used for classifying promoters can be further extended to classifying other parts of DNA, which can help with disease detection, as well as develop a fuller understanding of DNA. Therefore, developing an understanding of the complex function and detection of promoters proves a significant challenge in the field of bioinformatics. The accurate detection of these promoter sequences has significant implications for understanding the formation of proteins, a topic which is inextricably linked to disease detection and mitigation at a genetic level.

2.5 Exons and Introns

The second dataset utilized in this work includes detecting whether a given DNA sequence makes up an exon (coding) or intron (intergenomic) sequence. Understanding and detecting these two key facets of DNA comprise another challenge in modern bioinformatics. Prior to the process of translation (the translation of mRNA into proteins), certain sequences need to be spliced out of the mRNA sequence before the protein can be properly created. The sequences that remain are called exons, while the sequences that were removed are called introns. Note

that the mRNA sequence stems from the host's original DNA, in a process called transcription. As a result, predicting whether or not sequences are exons or introns when given only the DNA sequence aids with being able to predict the components and potential structure of a future protein. Ultimately, the dataset used is called the "Coding vs Intergenomic Sequences," or CVI dataset.

2.6 Threat Model

Our methodology aims to protect user data privacy while creating a gateway for neural network inference on encrypted nucleotide data. We assume an *honest-but-curious (HBC)* cloud that performs proper operations on encrypted data sent for inference, but is incentivized to extract as much information as possible from these operations. In addition, we consider the case of leakages during data transfer, the prevalence of which proves severely detrimental for plaintext federated inference.

Potential Attack Vector: A malicious actor manages to access the connection between a user and the global model during encrypted inference. Without homomorphic encryption, the malicious actor can access the data that the user sends, allowing them to have access to private genetic information.

Mitigation: Through the use of fully homomorphic encryption, these man-in-the-middle (MITM) attacks are preventable and trivial, as the attacker cannot decrypt the user's data unless they have access to the user's private key.

Additionally, we consider the idea of model poisoning, in which a malicious actor gains control of a local model during training and tries to pass faulty local weights. While our frameworks mainly focus on inference privacy, this attack vector during training, as well as membership inference attacks [29] can be trivialized through adjusting the federated aggregation scheme or utilizing differential privacy [11]. When testing our frameworks with FedMedian, an aggregation method that filters out anomalies that may stem from these unwanted actors, we found no significant change in model performance during training. This finding shows that our frameworks can easily be extended to create training-time security as well, without any significant model performance issues. [9]

3 Our Proposed Methodology

In our proposed framework, we assume that there are multiple local clients, and they train a global model with federated learning. Then, a user intends to send a DNA sequence to be evaluated homomorphically by the model. The global model receives and performs computations on this encrypted data, and then returns the results to the user, who can then decrypt them. Privacy and precision are ensured through the use of the TFHE and CKKS cryptosystem.

Fundamentally, all DNA sequences contain four nucleotides: Adenine (A), Thymine (T), Cytosine (C), and Guanine (G). A DNA sequence, therefore, can be expressed with a string with some combination of the letters [A,T,C,G], and this string serves as the elementary genetic code. However, these DNA sequences

can be extraordinarily long; consequently, understanding the meaning behind a string of DNA proves extraordinarily difficult. Moreover, with regard to machine learning, data centers often have limited amounts of data, making it difficult to train a single centralized machine learning model; therefore, a federated framework is a more effective framework for genomic data.

DNA proves to be a more challenging than previously accepted forms of data. Related works mainly used either numeric data, or data from SNPs, which are short strands of known variation in DNA. Meanwhile, the methodology proposed in this work targets longer strands of DNA (with our datasets having between up to 573 nucleotides per strand of DNA). Moreover, while certain related works which use both numeric and SNP data use logistic or linear regression [5, 8, 12], a classification model was used in this framework to ensure non-linear pattern recognition.

3.1 Data Preprocessing

To validate our approach, two main sources of DNA data were used. The two datasets stem from an open-access collection of datasets published in the Springer BMC Genomics Journal (n = 154k, up to 573 nucleotides per feature, and n = 100k, up to 200 nucleotides per feature) [17]. The data from this article is part of the Ensembl Genome Browser. We note that no synthetic data was generated to increase the size of the dataset, since given the cryptic nature of DNA, any synthetic generation could defy biological conventions. Our work requires multiple strands of labeled DNA sequenced data, from which we can discern the correlation between the nucleotides in the strand and the presence of a promoter sequence, or whether a certain strand includes an exon or intron. To preprocess the data, k-mer encoding was utilized. This separates a large sequence into groups of subgroups of k nucleotides, and stores them by frequency. In this work, we set k=4 to balance local sequential dependency with the noise or suppression of patterns which may stem from a larger k-value. As a result, this preprocessing method captures local dependencies within larger DNA strings. Then, StratifiedKFold was used to split the data across five local clients, while preserving the original class distribution across each local dataset. During the data split, a seed of 42 was used. Local models were solely given access to their local data to simulate an authentic use case for federated learning.

3.2 Model Architectures

In this work, we ran a variety of classical model types to demonstrate the feasibility of encrypted federated inference on DNA nucleotide data. First, a multilayer perceptron network (MLP) was utilized as the benchmark model. Moreover, we noted the changes in metrics that an LSTM model would provide. We note that for encrypted inference with the LSTM, knowledge distillation [18] into a smaller neural network was performed, as performing encrypted inference on an LSTM network proved computationally infeasible. Finally, a support vector machine

(SVM) was used as a lighter, less computationally intensive framework for comparison. A detailed description of the model sizes are shown in Table 1. The train-test split sizes were roughly either 80/20 (for the promoter dataset) or 75/25 (for the CVI dataset).

Table 1. Model Type vs Parameters and Global/Local Epochs Trained

	MLP	**LSTM (Distilled)**	**SVM**
Parameters	9313	54049	257
Global Epochs	25	25	20
Local Epochs	20	20	20

3.3 Federated Framework and Training

The local classification models were built using PyTorch, and then were compiled into FHE using the Concrete library. For the MLP model, the input layer contained 256 input neurons, the middle (hidden) layer contained 32 neurons, and the output layer included 2 neurons. Additionally, between each layer, there was a dropout layer, with dropout rate of 0.4; additionally, the ReLU function was utilized. The LSTM model contained three layers, and the SVM contained one layer.The federated classification training phase is performed in two steps: first, the local models are trained for a specified amount of time on the local data. In our framework, all local models trained for 20 local epochs, while the global model trained for 25 epochs; training past 25 global epochs showed negligible changes in our metrics. Then, the local models update the global model through weight aggregation by the federated average equation, which is expressed as follows:

$$w^{(t+1)} = \frac{1}{K} \sum_{k=1}^{K} w_k^{(t),} \tag{1}$$

where

- $w^{(t+1)}$: global model weights after iteration $t + 1$,
- $w_k^{(t)}$: model weights of client k after local training in iteration t,
- K: number of clients.

It should be noted that this equation is simplified on the assumption that the amount of data that each client receives is equal to the other clients; if the amount of data received was different, the formula would simply reflect a weighted average, rather than a normal average.

After the global model receives local weight updates, the parameters of the global model are redistributed to the local models, which then use those parameters in the next iteration. This continues for a specified number of global iterations. We note that the ReLU activation function [1] was specifically used

due to its lower computational cost during the future inference phase compared to other common activation functions such as the Sigmoid or TanH functions.

Prior to inference, the LSTM model had to undergo key changes to allow for computationally efficient inference. Particularly, an LSTM network makes pervasive use of the TanH or Sigmoid functions, which would need to be heavily approximated for encrypted inference. As a result, implementing an LSTM model which can perform encrypted inference proves infeasible both from a practicality standpoint and a computation standpoint. To mitigate this issue, we performed knowledge distillation [18] to develop a child model, which was an MLP. This MLP was trained using the LSTM's post-training weights, and was trained to ensure a similar accuracy to the original LSTM. For the knowledge distillation, a temperature of 3.0 was used. The temperature parameter is always positive, and represents the richness of the parent's (LSTM's) predictions when transferred to the child. A higher value for temperature represents a more even split of class probabilities, which allows the child to more effectively mimic the parent behavior.

3.4 Inference and Prediction

Fundamentally, inference occurs after the training and model weight compilation are finished. After a specified number of epochs, the training of the global model is complete. As training continues (as mentioned before), the weights are compiled to the global model, and the local models are updated with the new compiled weights. Therefore, after training is complete, all local clients will have a fully trained global model from a diverse dataset, even if they contributed a small amount of data on their own. This paradigm addresses the issue of low amounts of data within local clients; however, when data is compiled within multiple clients, a strong federated machine learning model can be created without divulging the data.

FHE Implementation Details: For the model prediction phase, the Concrete ML library [35] (version 1.9.0) for the TFHE cryptosystem was used. This framework is a Python wrapper for the TFHE scheme [7], which was originally written in Rust. Additionally, we utilized the TenSeal library for the CKKS cryptosystem, with a polynomial approximation degree of 16384. Both the TFHE and CKKS implementations ensure 128-bit security, and the key size was 236 MB. Finally, for the CKKS inference procedure, we used quadratic approximation for the ReLU function. The model was trained in plaintext, and then compiled for FHE using ConcreteML or TenSeal. The weights and biases were quantized at an 6 bit-width for the TFHE encryption rounds, with an approximation error of 0.01. The key size for TFHE was 568MB, which stems from the frameworks being 6-bit and 256 input neurons. The activation function was approximated using polynomials, and this allows the global model to be compiled and ready for inference on encrypted data. Figure 2 depicts the relationship between MLP inference accuracy and bit width.

The prediction phase of this framework was designed with the practical use case in mind: a client who intends to utilize our global model for classification,

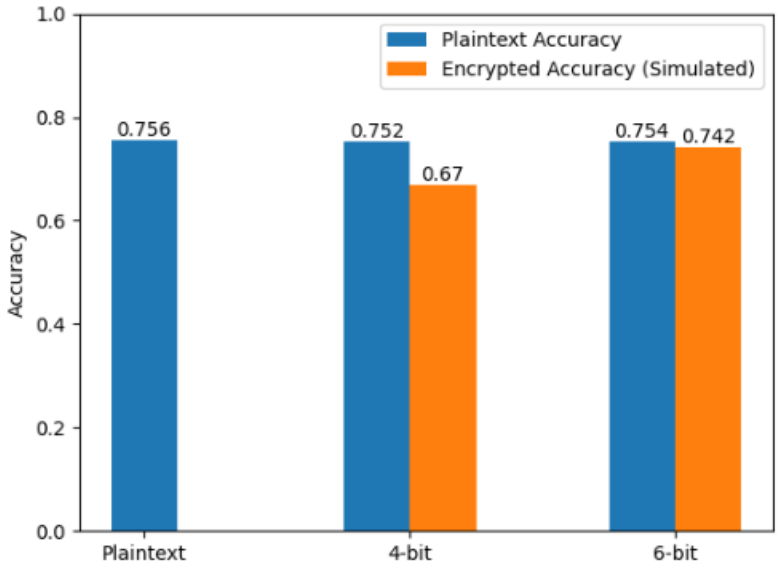

Fig. 2. MLP Inference Accuracy vs Bit Width: The MLP utilized for inference was trained for 15 global epochs, with 10 local epochs per global epoch. Then, encrypted inference was performed, and the accuracies of the models at different bit-widths were noted. At bit-widths larger than 6-bit, the requisite matrix multiplications exceed the size limitations of ConcreteML, making 6 bit width the optimal parameter for preserving accuracy, while also staying within the limits of computation.

but wants to keep the DNA data private. Therefore, FHE is used to encrypt the data before it goes into the global model for inference. The model is specifically designed to perform calculations on encrypted data and return the output. Since the data is encrypted homomorphically, the global server will have no access to it, preserving the data's privacy and preventing data transfer attacks. The inference process is summarized further in Algorithm 1. First, the user submits their encrypted data (depicted as δ, with the *nth* data strand being denoted as D_E^n), which they encrypted using their secret key. Then, the global model (which can perform computations homomorphically), performs inference and returns the result to the user, who can then use their key to decrypt back to plaintext. We note that *FHEeval* refers to the process of sending encrypted data to the global model for inference.

During the model inference phase, the decryption was done automatically; however, in a real-world use case, the final *if-statement* would apply, since the user would need to utilize their private key to decrypt *results*. In our framework, this encryption and decryption is completed automatically. Practically, the user that uploads a sequence to the global model for inference would have their own key, which would allow them to view *results*.

Algorithm 1 Prediction with FHE

Require: Encrypted data $\delta = (D_E^1, D_E^2, D_E^3 \ldots, D_E^n)$, FHE Circuit C
Ensure: The decrypted results from homomorphic inference
1: Initialize list *results*
2: **for** each $D_E^i \in \delta$ **do**
3: $\quad$ *results*$[i] \leftarrow$ FHEeval(C, D_E^i)
4: **end for**
5: **if** permission is given **then** ▷ Done automatically during inference
6: $\quad$ User uses private keys K to decrypt *results*
7: **end if**

4 Results and Discussion

This section presents the experimental data collected, along with details regarding the hardware used in our methodology. First, all of our experiments were performed on an Amazon EC2 c7i.8xlarge instance, which contained 32 vCPUs and 64 GiB of memory.

The scalability of our framework depends on the parameters of the framework, as noise growth of ciphertexts may present a larger issue as the number of clients or the length of data strings increase. While classical machine learning models result in a linear increase in complexity as input size increases, our framework contains a privacy-preserving facet, which causes noise to increase more quickly. As the input size of the data increases, the computational overhead increases at a significant rate, which presents a limitation of our work.

At a system level, the use of FHE means that users hold their own private keys, with which they encrypt their data before sending it to a model for evaluation. As a result, the keys are held by the users. Bootstrapping during inference is done automatically by the Concrete and TenSeal libraries, and happens at a per-layer granularity. Additionally, the offline costs include the creation of keys and the encrypting/decrypting of data, while the online costs, which includes the encrypted inference, are significantly higher. Finally, the encrypted inference with TFHE was conducted in a parallel fashion, with all of the 32 vCPU cores being utilized to speed up the inference process.

4.1 Experimental Evaluation

We developed a set of experiments that accurately captured the abilities of our methodology. First, we present the execution times with each model type in Table 2 and Table 3.

These tables demonstrate the significant decrease in computing time required for encrypted inference when utilizing the CKKS scheme compared to the TFHE scheme for both the LSTM and MLP models. Additionally, when the execution times above are broken into encoding, inference, and decoding, it proves evident that the major bottleneck is the encrypted inference, taking up the overwhelming majority of compute time.

Table 2. Execution Times (s) for Different Model Architectures and Encryption Types for the Promoter Dataset

	MLP	**LSTM (Distilled)**	**SVM**
TFHE	13658.33	13214	13.471
CKKS	488.23	487.29	165.45

Table 3. Execution Times (s) for Different Model Architectures and Encryption Types for the CVI Dataset

	MLP	**LSTM (Distilled)**	**SVM**
TFHE	14851.16	15405.07	13.965
CKKS	488.23	484.25	165.36

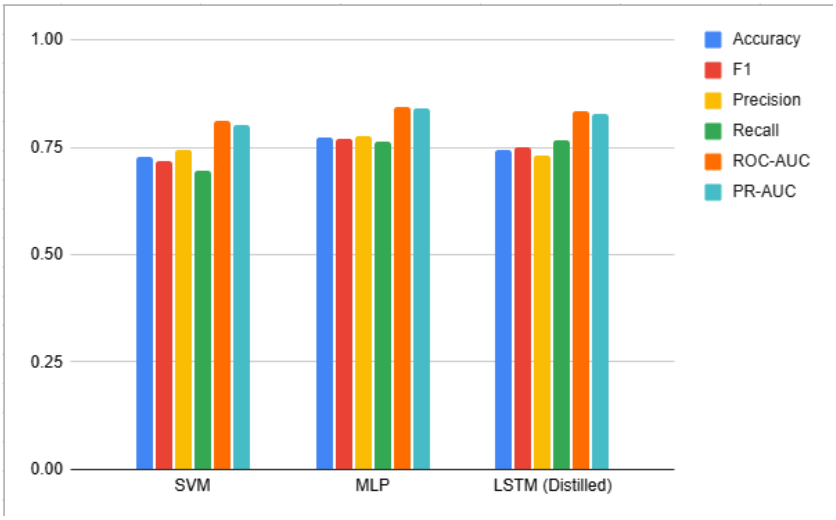

Fig. 3. Performance (Accuracy, F1, Precision, Recall, ROC-AUC, PR-AUC) vs Model Type for the Promoter Dataset and TFHE encryption

4.2 Results and Discussion

The performance of our frameworks were evaluated and compared through a variety of standard metrics: accuracy, F1 score, precision, recall, ROC-AUC, and PR-AUC. Additionally, confusion matrices of the models are shown in the Appendix. These metrics were chosen specifically to holistically evaluate each model during encrypted inference and post-training. Detailed graphs of model performance across these metrics are presented in Figure 3, which shows the models' performance when TFHE inference was conducted, and Figure 4, where CKKS inference was conducted. As shown in Figure 3 and 4, the LSTM and MLP perform similarly across most metrics; however, the SVM lags behind in every almost metric. These figures demonstrate that although lighter classical

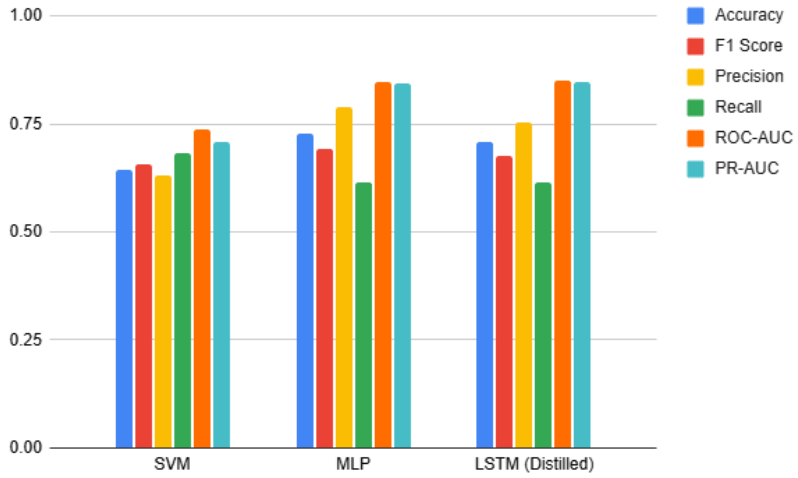

Fig. 4. Performance (Accuracy, F1, Precision, Recall, ROC-AUC, PR-AUC) vs Model Type for the Promoter Dataset and CKKS encryption

baselines can perform computations significantly faster, the heavier models are required to truly understand these complex sequences.

In addition, the disparity in performance between the Figure 3 and Figure 4 reflects the tradeoff between latency and accuracy: although the use of the CKKS scheme sped up model throughput for the MLP and LSTM, it resulted in a significantly lower performance.

A similar trend in performance is shown in the CVI dataset; however, the models performed significantly better across every metric. This increase in performance can likely be accounted for by the significant decrease in feature length, which reduced the overall complexity of the data. Model performances across varying models and varying cryptosystems are shown in Figures 5 and 6. The model performances with the CVI dataset also demonstrate similar trends to the promoter dataset, with the LSTM and MLP significantly outperforming the SVM. Our findings consequently demonstrate the importance of utilizing complex, nonlinear models for federated, encrypted inference on DNA nucleotide data. In addition, our findings show that the TFHE cryptosystem is optimal for accuracy, with a clear and significant decrease in performance when using the CKKS framework.

Notably, our model performance remained steady even through an increase in the number of clients, as shown in Figures 7 and 8. Finally, we compare our findings to related works. To our knowledge, our framework is the first privacy-preserving decentralized framework for nucleotide data. The capabilities of our project compared to select related works are shown in Table 4. We note that given the novel nature of our framework, none of the related works that

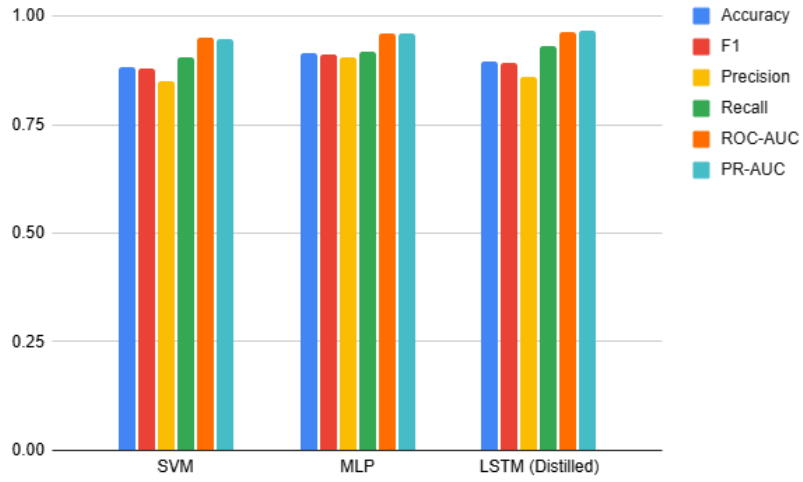

Fig. 5. Performance (Accuracy, F1, Precision, Recall, ROC-AUC, PR-AUC) vs Model Type for the CVI Dataset and TFHE encryption

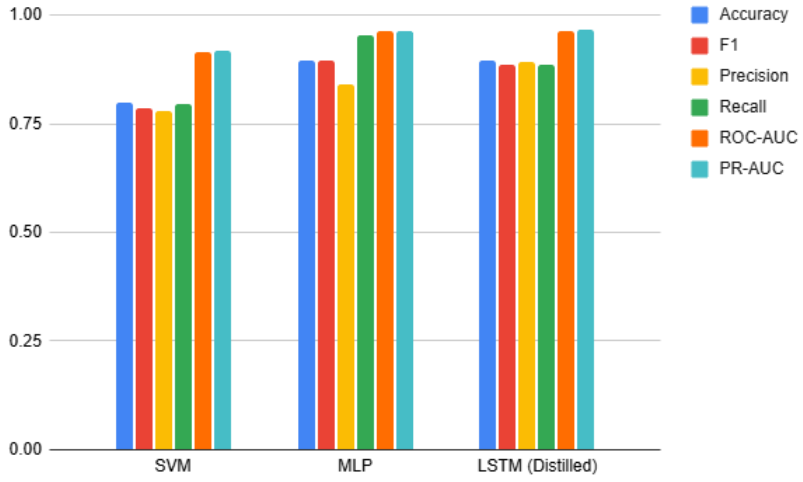

Fig. 6. Performance (Accuracy, F1, Precision, Recall, ROC-AUC, PR-AUC) vs Model Type for the CVI Dataset and CKKS encryption

employ decentralized learning and fully homomorphic encryption can predict with nucleotide data.

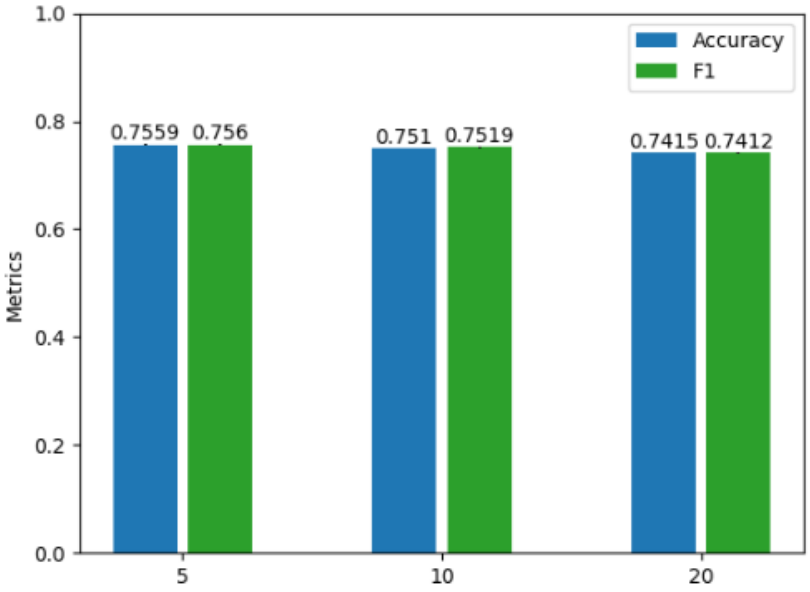

Fig. 7. Clients vs Model Accuracy: Effect of the number of clients on the final global model accuracy on the testing dataset after training for 15 global epochs and 10 local epochs with the larger promoter dataset. We note that the overall runtime slightly fluctuated in no discernible pattern with the increase in clients from 5 to 10 to 20.

4.3 Impact and Limitations

Our methodology demonstrates a novel approach to leveraging nucleotide data, which has become more ubiquitous in the field of biotechnology through DNA sequencing, in complex biomedical tasks. Our methodology not only utilizes the power of federated learning to generate a decentralized model that can be trained without sharing sensitive data but also capitalizes on the capabilities of fully homomorphic encryption to ensure data privacy during model inference. This approach allows for a truly privacy-preserving machine learning system, and to our knowledge, the first framework for predictions upon raw nucleotide data.

Our framework offers numerous advantages in the field of utilizing nucleotide data in machine learning contexts, as it creates an optimized pathway to train a decentralized model and perform encrypted computations without compromising on accuracy. However, some limitations of this project include its overall computational overhead. While our proposed framework offers promising accuracy, the use of FHE during prediction is impacting the inference throughput. Furthermore, in cases where leakage is possible during the training phase, our model may not be fully secure. Therefore, future extensions may focus on the use of differential privacy during the training phase to further reinforce the security

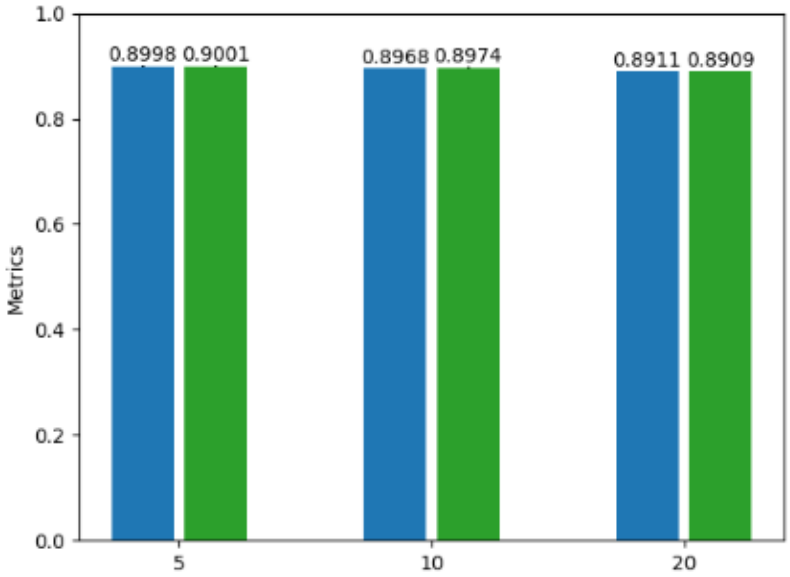

Fig. 8. Clients vs Model Accuracy: Effect of the number of clients on the final global model accuracy on the testing dataset after training for 15 global epochs and 10 local epochs with the CVI dataset. We note that the overall runtime slightly fluctuated in no discernible pattern with the increase in clients from 5 to 10 to 20. The legend remains the same as Figure 7.

of a privacy-preserving federated model. Another future direction can investigate extending our findings towards non-IID data. A major weakness of our work stems from its loss of accuracy and F1 when clients are presented with significantly imbalanced data for both the promoter and the CVI dataset. These findings are shown in Figures 9 and 10. Ameliorating these issues would greatly improve the scale that the findings in this paper can be used.

5 Related Works

In this section, we discuss previous works in the intersection between federated learning, fully homomorphic encryption, and medical data. These related works can be separated into three main classes: works focusing on imaging data, works focusing on numeric gene expression data, and works focusing on single-nucleotide polymorphisms. While these methods primarily focus on shorter strings of DNA (numeric or SNPs) or on imaging data, our work focuses on nucleotide-based data, which proves significantly more complex due to its importance on temporal dependency, as well as the inherent complexity of predicting on large strings of DNA, where understanding the meaning of the whole sequence stems

Table 4. Comparative Analysis of Privacy-Preserving Genomic Analysis Frameworks.

Proposed Framework	Federated Learning	Uses FHE	Predicts on DNA Nucleotides	Predicts on SNPs
This Work	✓	✓	✓	
Cho et al. [8]	✓	✓		✓
Froelicher et al. [12]	✓	✓		✓
Namazi et al. [24]	✓	✓		✓
Ananya et al. [2]			✓	
Umarov et al. [34]			✓	
Kapoor et al. [21]	✓		✓	

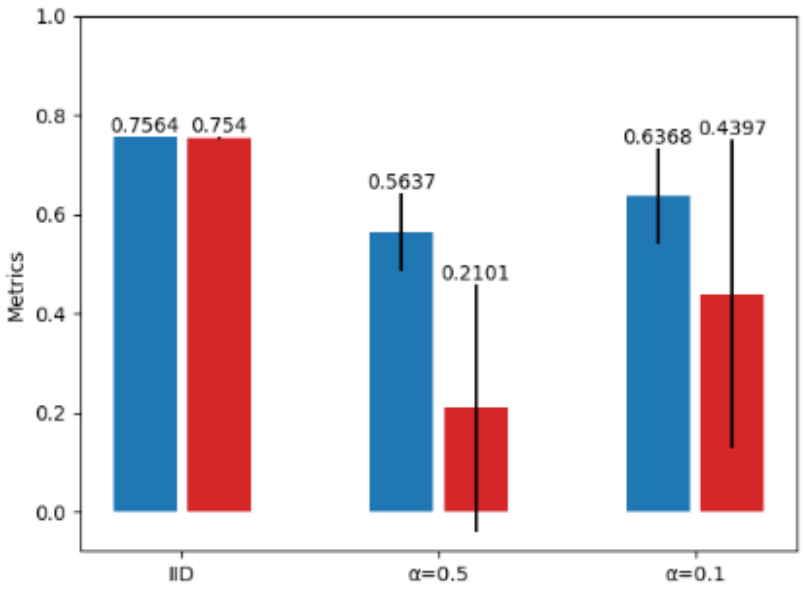

Fig. 9. Clients vs Model Accuracy: Effect of the class imbalance in local clients with constant training epochs, given by the Dirichlet α parameter, and the global model accuracy on the testing dataset after training with the promoter dataset. We note that the runtime strictly increased, from 277.3 seconds (IID) to 344.6 seconds ($\alpha = 0.5$) to 369.6 seconds ($\alpha = 0.1$).

from understanding the components that make it up. We discuss the capabilities of these works and compare them to our methodology.

While some related works have used DNA data for classification [2, 21, 34], these works do not incorporate fully homomorphic encryption during prediction, leaving privacy as a key concern. These works are omitted from the three prevailing classes that we focus on in this work.

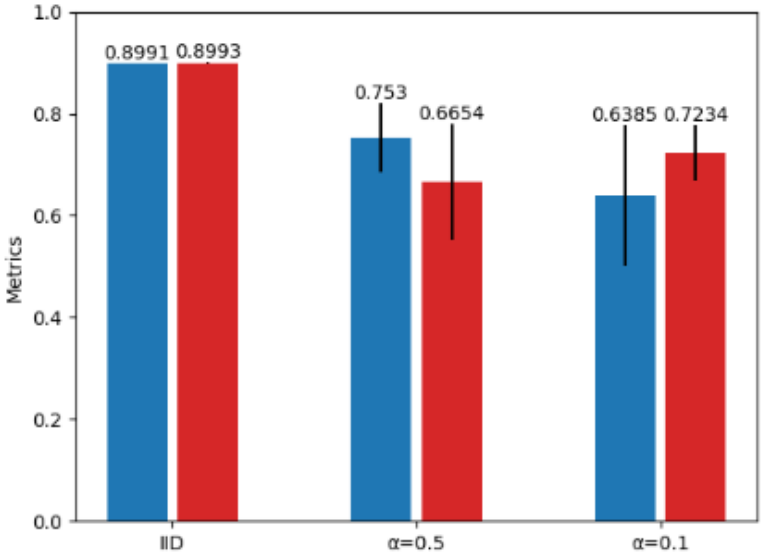

Fig. 10. Clients vs Model Accuracy: Effect of the class imbalance in local clients with constant training epochs, given by the Dirichlet α parameter, and the global model accuracy after training in the CVI dataset. We note that the runtime strictly increased, from 252.7 seconds (IID) to 283.5 seconds ($\alpha = 0.5$) to 339.9 seconds ($\alpha = 0.1$).

5.1 Class 1: Imaging Data

Multiple previous works exist regarding using FHE and FL schemes on imaging data [22, 32, 31, 30, 33]. These works mainly utilize the CKKS FHE framework for their computations, which utilizes an approximation-based approach to encryption. This approach increases the number of potential computations; however, it leads to reduced precision when encrypting input data. However, while images can be accurately represented despite minor pixel inaccuracies [19], DNA has an exceptionally low noise tolerance, since one small nucleotide change can result in a completely different output. Also, DNA sequences contain a sequential specificity that is intrinsic to determining their meaning, which images do not contain. This specificity requires added precision when encoding and performing operations on the data compared to images. Therefore, compared to these works, our methodology offers the same privacy guarantees during training (due to the decentralized nature of our framework) and inference (with 128 bits of security from FHE), but also (as shown by Figures 3 - 6) provides accurate predictions without compromising the data.

5.2 Class 2 and 3: Clinical Data and Single Nucleotide Polymorphisms

Many previous works focus on using numeric data to develop robust classification and regression models. For instance, Raisaro *et al.* [26] developed a framework which can predict off of clinical patient data, as well as from numeric gene expression data regarding the position of a mutated allele in a specific chromosome. This allowed them to develop a cancer prediction framework on both numeric data and clinical data. Similarly, Carpov *et al.* [5] used numeric gene expression data as well to develop a regression model. However, neither of these data types operate on large strings of genomic data, and rather perform computations on numeric data, which is significantly less complex. In this way, existing methodologies that operate on solely numeric data are inefficient and unusable for DNA nucleotide data. While these related frameworks prove more holistic, they fail to predict on data with as much complexity as long strings of DNA nucleotide data, which is the data type targeted in our framework.

Moreover, multiple related works utilize SNP data in their privacy-preserving frameworks [8, 12, 24, 28]. Single-nucleotide polymorphisms are specific nucleotides that have been altered in a given DNA sample. However, the data is often stored as either the few nucleotides that have been altered or as a numeric vector of the former. These mutations allow for predictive analytics regarding certain diseases and prove extremely useful to find the link between a given mutation and predisposition to a disease. However, these methodologies lack the ability to investigate and find meaning across long strings of DNA. Meanwhile, our methodology targets strings of DNA that are up to 573 nucleotides in length, in order to find biologically significant meaning. Although SNPs prove extremely useful, they are limited in scope; our work operates on DNA sequences that are much broader in scope and have more overall meaning to uncover.

5.3 Novel Chaotic and Memristive Systems and Their Potential Utility

While the scope of this work focuses mainly on FHE-based inference with federated machine learning, there exist other chaos and memristive-based systems that prove promising. These chaotic systems are excellent for generating pseudorandom numbers, which lend themselves to cryptography. In this way, these systems can be used for encrypting genomic sequences and preserve security. The promise that these systems display for strong encryption make it a valuable methodology for encrypting genomic sequences similar to those presented in the paper [13]. While the methodology shown in [13] and [14] demonstrates the ability of these chaotic systems to encrypt images, this functionality can easily be extended to genomic sequences as an alternative to FHE.

6 Conclusion

In this work, we present a novel framework for privacy-preserving decentralized machine learning with nucleotide data. Previous works developed similar models

for imaging and numeric-based data; however, this ignores the rising prevalence of nucleotide-based data. In contrast to images and numeric data, nucleotide data is sequentially sensitive, as one misrepresentation can lead to a completely different output from the DNA. By utilizing the TFHE cryptosystem to ensure precise data encryption, we can create an accurate framework that can be deployed in real-world contexts. Additionally, we compare our methodologies in both TFHE and CKKS, and demonstrate that although CKKS computations are faster on average, this speed comes at the cost of accuracy. We demonstrate the performance of our three models in two highly critical genomic settings: identifying promoters, and identifying coding or intergenomic sequences (exons and introns). Our work paves the way for future research into using collaborative and encrypted machine learning to predict on complex DNA sequences.

As DNA sequencing gains popularity in the field of biotechnology, predictive analytics on nucleotide data proves notably more important. Our framework depicts the ability to embrace this new method of capturing DNA and engage with it in a meaningful way. Our methodology depicts the ability to use DNA information to accurately decrypt its hidden code, which can in the future be applied to disease detection and further understand the meaning behind a given DNA strand.

Appendix

In this appendix, we present the confusion matrices for each model when training is complete.

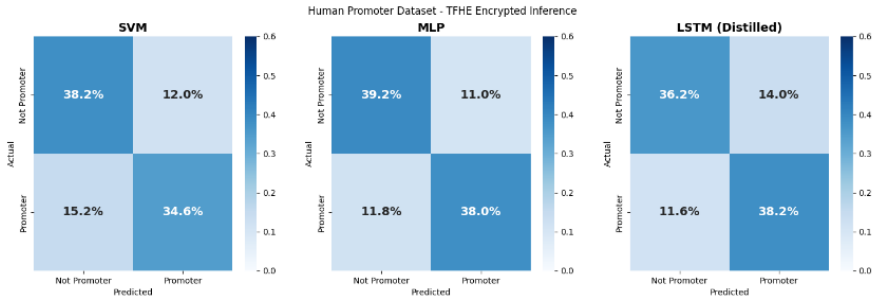

Fig. 11. Confusion Matrices for the SVM, MLP, and LSTM (Distilled) Architectures with the Promoter Dataset and TFHE encryption.

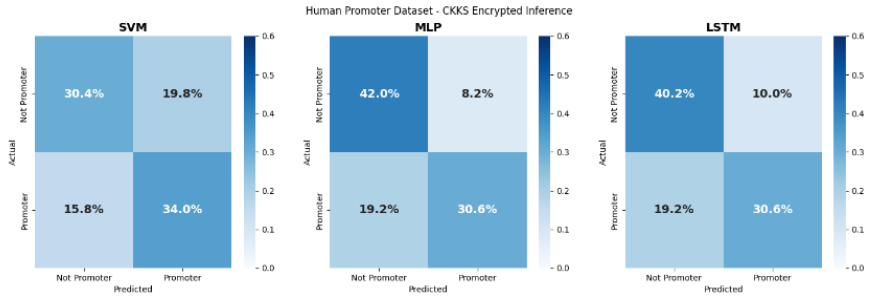

Fig. 12. Confusion Matrices for the SVM, MLP, and LSTM (Distilled) Architectures with the Promoter Dataset and CKKS encryption.

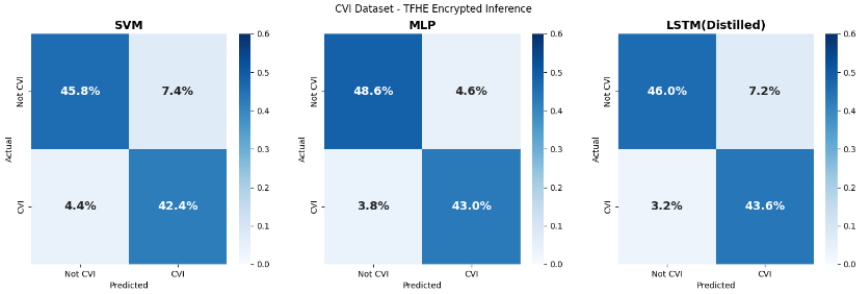

Fig. 13. Confusion Matrices for the SVM, MLP, and LSTM (Distilled) Architectures with the CVI Dataset and TFHE encryption.

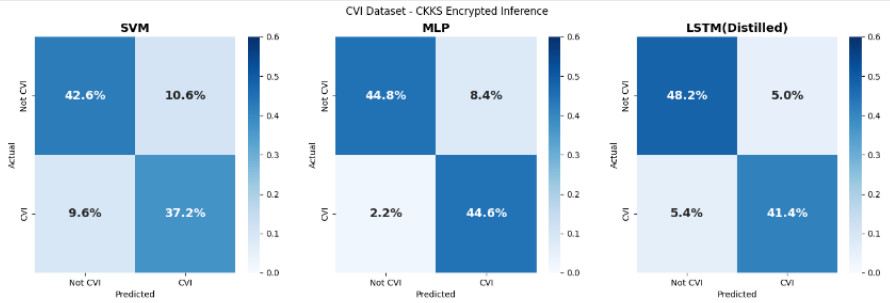

Fig. 14. Confusion Matrices for the SVM, MLP, and LSTM (Distilled) Architectures with the CVI Dataset and CKKS encryption.

References

[1] Abien Fred Agarap. *Deep Learning using Rectified Linear Units (ReLU)*. 2019. arXiv: 1803.08375 [cs.NE]. URL: https://arxiv.org/abs/1803.08375.

[2] Namburi Ananya et al. "DNA Classification For Finding E.coli". In: *2024 3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*. 2024, pp. 1762–1768. DOI: 10.1109/ICAAIC60222.2024.10575339.

[3] Nathalie Baracaldo and Hayim Shaul. *Federated learning meets Homomorphic encryption*. Jan. 2023. URL: https://research.ibm.com/blog/federated-learning-homomorphic-encryption.

[4] Keith Bonawitz et al. "Practical Secure Aggregation for Privacy-Preserving Machine Learning". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. CCS '17. Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 1175–1191. ISBN: 9781450349468. DOI: 10.1145/3133956.3133982. URL: https://doi.org/10.1145/3133956.3133982.

[5] Sergiu Carpov et al. "GenoPPML – A framework for genomic privacy-preserving machine learning". In: *IEEE 15th International Conference on Cloud Computing (CLOUD)* (July 2022), pp. 532–542. DOI: 10.1109/cloud55607.2022.00076.

[6] Jung Hee Cheon et al. "Homomorphic encryption for arithmetic of approximate numbers". In: *Lecture Notes in Computer Science* (2017), pp. 409–437. DOI: 10.1007/978-3-319-70694-8_15.

[7] Ilaria Chillotti et al. *Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds*. Feb. 2017. URL: https://eprint.iacr.org/2016/870.

[8] Hyunghoon Cho et al. "Secure and federated genome-wide association studies for biobank-scale datasets". In: *Nature Genetics* (2025), pp. 1–6.

[9] Judith Sáinz-Pardo Díaz and Álvaro López García. *Enhancing the Convergence of Federated Learning Aggregation Strategies with Limited Data*. 2025. arXiv: 2501.15949 [cs.LG]. URL: https://arxiv.org/abs/2501.15949.

[10] Prerna Dogra. *Federated learning with flare: Nvidia brings collaborative AI to healthcare and beyond*. Nov. 2022. URL: https://blogs.nvidia.com/blog/federated-learning-ai-nvidia-flare/.

[11] Cynthia Dwork and Aaron Roth. "The Algorithmic Foundations of Differential Privacy". In: *Foundations and Trends in Theoretical Computer Science* 9.3-4 (Aug. 2014), pp. 211–487. ISSN: 1551-305X. DOI: 10.1561/0400000042. eprint: https://www.emerald.com/fttcs/article-pdf/9/3-4/211/11486280/0400000042en.pdf. URL: https://doi.org/10.1561/0400000042.

[12] David Froelicher et al. "Truly privacy-preserving Federated Analytics for precision medicine with multiparty homomorphic encryption". In: *Nature Communications* 12.1 (Oct. 2021). DOI: 10.1038/s41467-021-25972-y.

[13] Suo Gao et al. "A 3D Memristive Cubic Map With Dual Discrete Memristors: Design, Implementation, and Application in Image Encryption".

In: *IEEE Transactions on Circuits and Systems for Video Technology* 35.8 (2025), pp. 7706–7718. DOI: 10.1109/TCSVT.2025.3545868.

[14] Suo Gao et al. "Encrypt a Story: A Video Segment Encryption Method Based on the Discrete Sinusoidal Memristive Rulkov Neuron". In: *IEEE Transactions on Dependable and Secure Computing* 22.6 (2025), pp. 8011–8024. DOI: 10.1109/TDSC.2025.3603570.

[15] Craig Gentry. "Fully homomorphic encryption using ideal lattices". In: *Proceedings of the forty-first annual ACM symposium on Theory of computing* (May 2009), pp. 169–178. DOI: 10.1145/1536414.1536440.

[16] Charles Gouert and Nektarios Georgios Tsoutsos. "Data Privacy Made Easy: Enhancing applications with Homomorphic encryption". In: *ACM Transactions on Design Automation of Electronic Systems* 30.3 (Feb. 2025), pp. 1–31. DOI: 10.1145/3715877.

[17] Katarína Grešová et al. "Genomic benchmarks: a collection of datasets for genomic sequence classification". In: *BMC Genomic Data* 24.1 (2023), p. 25.

[18] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. 2015. arXiv: 1503.02531 [stat.ML]. URL: https://arxiv.org/abs/1503.02531.

[19] Hossein Hosseini, Sreeram Kannan, and Radha Poovendran. "Dropping Pixels for Adversarial Robustness". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2019, pp. 91–97. DOI: 10.1109/CVPRW.2019.00017.

[20] National Human Genome Research Institute. *DNA Sequencing Fact Sheet*. URL: https://www.genome.gov/about-genomics/fact-sheets/DNA-Sequencing-Fact-Sheet.

[21] Ayshika Kapoor et al. "iProLSTM-FL: A Federated Learning Framework for Promoter Identification using LSTM Networks". In: *International Conference on Signal Processing, Computing and Control (ISPCC)*. 2025, pp. 221–226. DOI: 10.1109/ISPCC66872.2025.11039561.

[22] Xavier Lessage et al. "Secure federated learning applied to medical imaging with fully homomorphic encryption". In: *IEEE International Conference on AI in Cybersecurity (ICAIC)*. 2024, pp. 1–12. DOI: 10.1109/ICAIC60265.2024.10433836.

[23] Pengrui Liu, Xiangrui Xu, and Wei Wang. "Threats, attacks and defenses to Federated Learning: Issues, taxonomy and Perspectives". In: *Cybersecurity* 5.1 (Feb. 2022). DOI: 10.1186/s42400-021-00105-6.

[24] Mina Namazi et al. "Privacy-preserving framework for genomic computations via multi-key homomorphic encryption". In: *Bioinformatics* 41.3 (Jan. 2025), btae754. ISSN: 1367-4811. URL: https://doi.org/10.1093/bioinformatics/btae754.

[25] Pascal Paillier. "Public-key cryptosystems based on composite degree residuosity classes". In: *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT'99. Prague, Czech Republic: Springer-Verlag, 1999, pp. 223–238.

[26] Jean Louis Raisaro et al. "MEDCO: Enabling secure and privacy-preserving exploration of distributed clinical and Genomic Data". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 16.4 (July 2019), pp. 1328–1341. DOI: 10.1109/tcbb.2018.2854776.

[27] R. L. Rivest, A. Shamir, and L. Adleman. "A method for obtaining digital signatures and public-key cryptosystems". In: *Communications of the ACM* 21.2 (Feb. 1978), pp. 120–126. DOI: 10.1145/359340.359342.

[28] Md Nazmus Sadat et al. "SAFETY: Secure gwAs in Federated Environment through a hYbrid Solution". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 16.1 (2019), pp. 93–102. DOI: 10.1109/TCBB.2018.2829760.

[29] Reza Shokri et al. "Membership Inference Attacks Against Machine Learning Models". In: *2017 IEEE Symposium on Security and Privacy (SP)*. 2017, pp. 3–18. DOI: 10.1109/SP.2017.41.

[30] Dimitris Stripelis et al. "A federated learning architecture for secure and private neuroimaging analysis". In: *Patterns* 5.8 (2024).

[31] Dimitris Stripelis et al. "Secure federated learning for neuroimaging". In: *arXiv preprint arXiv:2205.05249* 11 (2022).

[32] Dimitris Stripelis et al. "Secure neuroimaging analysis using federated learning with homomorphic encryption". In: *17th International Symposium on Medical Information Processing and Analysis*. Ed. by Letícia Rittner et al. Vol. 12088. International Society for Optics and Photonics. SPIE, 2021. URL: https://doi.org/10.1117/12.2606256.

[33] Daniel Truhn et al. "Encrypted federated learning for secure decentralized collaboration in cancer image analysis". In: *Medical Image Analysis* 92 (2024), p. 103059. ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2023.103059.

[34] Ramzan Umarov et al. "Promoter analysis and prediction in the human genome using sequence-based deep learning models". In: *Bioinformatics* 35 (Jan. 2019). DOI: 10.1093/bioinformatics/bty1068.

[35] Zama.ai. URL: https://docs.zama.ai/concrete-ml.